Engineering Simplicity

DAY ONE: IPSEC VPN COOKBOOK 2018

Find the right IPsec VPN solution in this cookbook full of use cases, sample configurations, and security best practices.

By Johan Andersson

DAY ONE: IPSEC VPN COOKBOOK 2018

In a world of evolving security threats and increasingly strict compliance, all organizations must ensure that data over a wide area network or the public Internet is secure and encrypted. But because there are so many options available for IPsec VPNs, it has always been difficult for architects and administrators to understand what functions to choose.

Day One: IPsec VPN Cookbook 2018 has a simple approach – pick one of the use case examples from each chapter, from basic to complex, and start implementing an IPsec VPN solution. That's why this cookbook has been one of the most popular internal Juniper documents for years, used by field sales and system engineers, and why it is now intended to have annual updates.

"Johan Andersson, one of the encryption wizards at Juniper Networks, does an amazing job of explaining the many IPsec VPN design choices and technologies by showing the reader exactly how to configure each one with the Junos OS. Highly recommended."

Bhupen Mistry, Security Architect & Consultant, Operativity Ltd.

IT'S DAY ONE AND YOU HAVE A JOB TO DO, SO LEARN HOW TO:

- Define the different types of IPsec VPN architectures and topologies.
- Choose the best IPsec VPN technology to for your use.
- Know the issues surrounding the integration of encryption technology.
- Understand Juniper's IPsec VPN solutions.
- Learn how to incorporate the IPsec VPN features into a design.
- Explore the steps required to implement and tune IPsec VPN in your network.
- Use architectural blueprint designs as a base template for your specific scenario.
- Understand the differences between the various types of IPsec VPNs.
- Understand and configure IPsec VPN features including security best practices.
- Verify your configuration using basic troubleshooting commands.
- Monitor the status of IPsec VPN across your network in real time.

ISBN 978-1-941441-71-8
9 781941 441718 53500

Day One: IPsec VPN Cookbook 2018

by Johan Andersson

JUNIPER NETWORKS

Published by Juniper Networks Books
Author: Johan Andersson
Technical Reviewers: Bhupen Mistry
Editor in Chief: Patrick Ames
Copyeditor: Nancy Koerbel

ISBN: 978-1-941441-71-8 (print)
Printed in the USA by Vervante Corporation.
ISBN: 978-1-941441-72-5 (ebook)

Version History: v1, June 2018
2 3 4 5 6 7 8 9 10

http://www.juniper.net/dayone

About the Author

Johan Andersson has over 18 years experience in Networking and IT Security within the automobile, pharmaceutical, finance, and service provider sectors. He has had multiple roles as a Network Engineer, Consultant, and Architect, and started with NetScreen Technologies as a Certified Instructor. Today, Johan is the IPsec Product Manager for Juniper Networks.

Feedback? Comments? Error reports? Email them to dayone@juniper.net.

How to Use This Cookbook

This book is organized into five chapters with many recipes inside each chapter:

- Chapter 1: Choose a IPsec VPN Design
- Chapter 2: Configure Authentication: PKI, Radius, and EAP-TLS
- Chapter 3: Pick a Site-to-Site Configuration
- Chapter 4: Configure ADVPN or GroupVPNv2
- Chapter 5: Deploy Remote Access VPN

Choose what you need in each recipe and then move on to the next recipe. You don't have to read each recipe in full, unless you want to know the differences between the included use cases.

Given the nature of IPsec VPNs and its many iterations, there are most certainly things missing in this collection of tutorials and IPsec VPN cookbook recipes. Not every possibility or scenario could be included, but given the 2018 time stamp on the title , this cookbook is intended to be updated with more material and new technology changes in forthcoming editions.

This Cookbook's Landing Page

Depending on when you are reading this cookbook, there may be updated versions or errata that has been corrected. Check this cookbook's landing page for updates, posts by other readers, and new versions. Also, you can leave posts for the author and editors here, on what's missing, what you would like to see, comments and critiques: https://forums.juniper.net/t5/Day-One-Books/Day-One-IPsec-VPNs-Cookbook-2018/ba-p/326916.

MORE? For setting up your SRX Series try using *Day One: SRX Series Up and Running with Advanced Security Services*: https://www.juniper.net/us/en/training/jnbooks/day-one/srx-up-running/index.page.

MORE? The Juniper TechLibrary's documentation on IPsec VPN deployment is thorough and starts here: https://www.juniper.net/documentation/en_US/junos-space17.2/topics/concept/junos-space-ipsec-vpn-overview.html.

Welcome to Day One

This cookbook is part of the *Day One* library, produced and published by Juniper Networks Books.

Day One books were conceived to help you get just the information that you need on day one. The series covers Junos OS and Juniper Networks networking essentials with straightforward explanations, step-by-step instructions, and practical examples that are easy to follow.

You can obtain publications from either series in multiple formats:

- Download a free PDF edition at http://www.juniper.net/dayone.
- Get the ebook edition for iPhones and iPads from the iBooks Store. Search for *Juniper Networks Books* or the title of this book.
- Get the ebook edition for any device that runs the Kindle app (Android, Kindle, iPad, PC, or Mac) by opening your device's Kindle app and going to the Amazon Kindle Store. Search for *Juniper Networks Books* or the title of this book.
- Purchase the paper edition at Vervante Corporation (www.vervante.com) for between $15-$40, depending on page length.
- Note that most mobile devices can also view PDF files.

What You Need to Know Before Reading This Cookbook

Before reading this cookbook, you should have a basic understanding of the Junos OS. Specifically, you should be able to change configurations and to navigate through the command line or graphical user interface hierarchy. You can reference other *Day One* books and online training to help you acquire this background. The *Day One* library can be found at: https://www.juniper.net/us/en/training/jnbooks/. Juniper Networks documentation can be found at: https://www.juniper.net/documentation.

After Reading This Cookbook, You'll Be Able To:

- Define the different types of IPsec VPN architectures and topologies.
- Choose the best IPsec VPN technology to for your use.
- Know the issues surrounding the integration of encryption technology.
- Understand Juniper's IPsec VPN solutions.
- Learn how to incorporate the IPsec VPN features into a design.

- Explore the detailed steps required to implement and tune IPsec VPN in your network.
- Use architectural blueprint designs as a base template for your specific scenario.
- Understand the differences between the different types of IPsec VPNs.
- Understand and configure IPsec VPN features including security best practices.
- Verify your configuration using basic troubleshooting commands.
- Monitor the status of IPsec VPN across your network in real time.

Statements and Definitions

Static External IP: This means that the IP address of the external interface configured as IKE peer is not changing.

Dynamic External IP: This means that the IP address of the external interface configured as IKE peers is dynamicly assigned through a protocol, like DHCP or PPPoE.

IKE version 1: IKEv1 is the legacy version of IKE, and is still the most commonly used version within Enterprises.

IKE version 2: Internet Key Exchange Version 2 is the next generation standard for secure key exchange between peer devices, defined in RFC 4306. IKEv2 is available in this release for securing IPsec traffic.

The advantages of using version 2 over version 1 are as follows:

- Simplifies existing IKEv1
- Single RFC, including NAT-T, EAP, and remote address acquisition
- Replaces the eight initial exchanges with a single four message exchange
- Reduces the latency for the IPsec SA setup and increases connection establishment speed
- Increases robustness against DOS attack
- Improves reliability through the use of sequence numbers, acknowledgments, and error correction
- Forward Compatibility
- Simple cryptographic mechanisms

Traffic selector negotiation:

- IKEv1: Responder can just say yes/no

- IKEv2: Negotiation ability added

Reliability

- All messages are request/response.
- Initiator is responsible for retransmission if it doesn't receive a response.

IKEv2 includes support for:

- Route-based VPN
- Site-to-site VPN
- Dead peer detection (liveness check)
- Chassis cluster
- Certificate-based authentication
- Hardware offloading of the ModExp operations in a Diffie Hellman (DH) exchange

Traffic selectors: An IKEv2 traffic selector is essentially the same as an IKEv1 proxy-ID. Traffic selectors and proxy-IDs are used the same way. IKEv2 specifies single traffic selector in each direction.

An IKEv2 child security association (SA) is known as a Phase 2 SA in IKEv1. The child SA differs in behavior from the Phase 2 SA in the following ways:

- IKE and child SA rekeying—In IKEv2, a child SA cannot exist without the underlying IKE SA. If a child SA is required, it will be rekeyed; however, if the child SAs are currently active, the corresponding IKE SA will be rekeyed.
- Version 1 and version 2

IKE external interface in VR: This means you can place the external IKE peer interface in a non-default virtual router.

Dead peer detection: Dead peer detection (DPD) is a method that network devices use to verify the current existence and availability of other peer devices. You can use DPD instead of VPN monitoring. However, you cannot use both features simultaneously. VPN monitoring applies to an individual IPsec VPN, while DPD is configured only in an individual IKE gateway context. A device performs DPD verification by sending encrypted IKE Phase 1 notification payloads (R-U-THERE messages) to a peer and waiting for DPD acknowledgements (R-U-THERE-ACK messages) from the peer. The device sends an R-U-THERE message only if it has not received any traffic from the peer during a specified DPD interval. If the device receives an R-U-THERE-ACK message from the peer during this interval, it considers the peer alive. If the device receives traffic on the tunnel from the peer, it

resets its R-U-THERE message counter for that tunnel, thus starting a new interval. If the device does not receive an R-U-THERE-ACK message during the interval, it considers the peer dead. When the device changes the status of a peer device to be dead, the device removes the Phase 1 security SA and all Phase 2 SAs for that peer.

The following DPD modes are supported on SRX Series devices:

- *Optimized:* R-U-THERE messages are triggered if there is no incoming IKE or IPsec traffic within a configured interval after the device sends outgoing packets to the peer. This is the default mode.
- *Probe idle tunnel*: R-U-THERE messages are triggered if there is no incoming or outgoing IKE or IPsec traffic within a configured interval. R-U-THERE messages are sent periodically to the peer until there is traffic activity. This mode helps in early detection of a downed peer and makes the tunnel available for data traffic.
- *Always send*: R-U-THERE messages are sent at configured intervals regardless of traffic activity between the peers.

NOTE It is recommended that the probe idle tunnel mode be used instead of the always-send mode.

VPN monitor: VPN monitoring is a Junos OS mechanism that monitors only Phase 2 SAs. VPN monitoring is enabled on a per-VPN basis with the VPN-monitor statement at the [edit security ipsec vpn vpn-name] hierarchy level. The destination IP and source interface must be specified. The optimized option enables the device to use traffic patterns as evidence of peer liveliness; ICMP requests are suppressed. VPN monitoring options are configured with the VPN-monitor-options statement at the [edit security ipsec] hierarchy level. These options apply to all VPNs for which VPN monitoring is enabled. Options you can configure include the interval at which ICMP requests are sent to the peer (the default is 10 seconds) and the number of consecutive ICMP requests sent without receiving a response before the peer is considered unreachable (the default is 10 consecutive requests)

Authentication PSK (Preshared Keys): Preshared keys are used to authenticate the IKE peer, the preshared key is a shared secret between two parties. If someone can figure out the key, then they also can access the information, as they can open the lock. Today it's very common for organizations to use preshared keys because they're simple and quick to implement. The problem is that most of these organizations also use the same key for all of their IPsec VPNs, which means you have to change all the preshared keys if you get compromised keys.

Authentication Cert: Certificate-based authentication for IKE peers uses an X.509 certificate which can be seen as a passport that can either be verified to be authentic or revoked. The owner of the certificate should, as soon as they are aware of a compromised certificate, report it to the administrator of the PKI system, this will then be revoked and should no longer be granted access when trying to establish a new tunnel. For this tunnel to come back online, the owner has to enroll a new certificate for this peer, and then the remote peer can verify that the access request is trusted and valid.

NAT inside tunnel: This is used when you have overlapping subnets on both ends of the tunnel, otherwise the client can't reach the other network.

NAT-T for IKE peer: NAT-T is used when there is a network device between the two tunnel end-points that enforce NAT.

ST interface in VR: In some cases you may want to add the ST interface in a customer defined routing instance instead of using the default routing instance.

Dynamic routing RIP: Instead of using static routing that points to a tunnel, you can use RIP as a dynamic routing protocol to find networks on the other side of the tunnel.

Dynamic routing OSPF: Instead of using static routing that points to a tunnel, you can use OSPF as a dynamic routing protocol to find networks on the other side of the tunnel.

Dynamic routing BGP: Instead of using static routing that point to a tunnel, you can use BGP as a dynamic routing protocol to find networks on the other side of the tunnel.

Chapter 1

Choose an IPsec VPN Design

This first chapter provides an overview of several IPsec VPN topologies offered by Juniper Networks, and when they can be used for site-to-site communication. Choose the design you need for your organization from the Topology Matrix.

Site-to-site IPsec VPN means that you can communicate securely between two or more areas by encrypting and authenticating the data transfer across a network. It can be used in any IP-based network as long as all devices that constitute the network allow IP protocol 50 Encapsulating Security Payload (ESP) and User Datagram Protocol (UDP) or UDP-encapsulated ESP packets.

NOTE Not all IPsec implementations support IPv4 or IPv6 and/or UDP-encapsulated ESP traffic.

Juniper has traditional IPsec VPNs that allow traffic to flow between two tunnel endpoints (site-to-site), IPsec VPNs that can also reroute traffic into another tunnel allowing multiple sites to communicate (hub-and-spoke), and IPsec VPNs that allow multiple sites to communicate with each other by setting up tunnels between all possible sites (full mesh). Both hub-and-spoke and full mesh topologies have drawbacks. Hub-and-spoke topologies can saturate the hub bandwidth when all traffic has to traverse the hub. This can introduce latency and jitter into communications as normal traffic patterns will vary over time. For full mesh topologies, the main drawback is the complication of managing tunnels, as you need to reconfigure every device as soon there is a new device, or a device that should be removed. To solve the full mesh challenges, a semi mesh topology was introduced, which allows a site to dynamically set up a tunnel to another site on-demand after an notification from the hub.

Because these topologies normally demand configuration of each endpoint when you need to install or remove another endpoint, Juniper has other solutions. Auto Discovery VPN (ADVPN) provides zero touch configuration on the central endpoint and minimal configuration on the other endpoints, but also allows you to set up dynamic tunnels between spokes without saturating the hub site.

There is also a topology called Group of Domain Interpretation (GDOI) that offers a so called full mesh deployment because of its one major limitation: GDOI requires a routable IP address as this topology does not encapsulate the source and destination IP address. While in practice it is only used internally or over MPLS networks, it's good to remember that the internal IP network scheme is exposed with GDOI.

The most important questions to answer regarding choosing an IPsec topology are:

- What kind of network will be used for the transport?
- Do you need a dynamic routing topology between sites?
- How much third-party connectivity do you need? (not all topologies work smoothly with all vendors)

Design Considerations

Once you have determined that you need to protect the data path when traffic flows between different sites, you need to understand the advantages and the disadvantages that each of the different IPsec VPN technologies provide, because there may be multiple ways to accomplish the same goal. For example, you could be running dynamic routing in your network, you could have a dynamically-assigned IP address on the external interface, or there might be a Network Address Translation (NAT) device between the sites. These are just a few scenarios you need to consider when planning your IPsec VPN topology.

The following design considerations cover challenges that you need to consider before deciding on your topology.

Public or Private Transport Environment (Internet or MPLS)

First decide if you have a public or private transport need. Depending on the need, you can choose either Internet or MPLS, or both, as the standard transport service. Remember that MPLS is not more secure than the Internet – it's just more isolated, which means it's more difficult to gain access to its transport network.

Every network is unique and each has different transport needs for its IP environment. Some networks may want the cheapest bandwidth possible, as they do not have any latency-critical applications; others may have latency-critical applications. Some have a need for communication with every other site or have other demands that make them choose a private MPLS transport over a public Internet transport, or vice versa. The most important thing here is that you know the IPsec topology that best fulfills your demands. (The Architectures and Topologies section later in this chapter describes each topology and which topology best fits each environment, followed by the cookbook's step-by-step integration guide.)

Static or Dynamic External IPs

Deciding on an IPsec VPN topology depends on your external interface IP address assignment. The main challenge with a dynamic IP is that the remote peer may not know how to find its tunnel endpoint. Some vendors do not support a FQDN address for the tunnel target.

As long as the responding peer has a fixed IP, it's normally not a problem. But if the responding peer has a dynamic IP address, you need a way to find that peer. The best way is to have a dynamic DNS service.

It's recommended that at least the responding peer (normally the central site) of the tunnel be static. Even if it's possible to establish a tunnel when all external interfaces have dynamic assignments, it's likely that you will later face challenges, especially with third-party tunnels, or a DNS service that is not working as it should.

Network Address Translation (NAT)

NAT can be used for two scenarios:

- The most common scenario is that there is a device between both IPsec endpoints that translates IP addresses. Some of these devices can't handle IPsec very well, and some can't handle it at all. So you may find some problems with your tunnels, but in most cases there is a workaround. With IKEv1, there are some challenges on how to configure this. IKEv2 solves this in the background. Of course, if one endpoint is hidden by a NAT device, you need to understand what IP address will be used to translate to the real IP of the endpoint. If the NAT device is not managed by your organization, you need to contact the administrator to find out what IP should be used to find your device behind the NAT device.
- The second challenge is when you have two sites with the same IP network, meaning a client on one site will only send broadcasts when trying to communicate with a device in the remote network. For packets to reach the gateway, you need to build a design that can solve this problem. The only way to do that is to set up a fake network that each client can connect to; the fake IP address will NAT your traffic to the other side. This can be done on either end of the tunnel. This option might not be supported by other vendors, which is why you should do this on the Junos side.

Dynamic or Static Routing Environments

No doubt you have different routing environments: static routing, dynamic protocols, or a mix of static and dynamic routing.

As different IPsec VPN technologies support different routing protocols, you need to keep this in mind when choosing your IPsec topology. Juniper Networks SRX Series devices support both dynamic and static routing when running route-based IPsec VPN. For AutoVPN, Juniper supports Auto Route Insertion and dynamic routing, and for ADVPN, it currently only supports the OSPF dynamic protocol.

Communication with Third Parties

When setting up IPsec VPNs with non-Juniper parties, you should always consider what those vendors are capable of, as there are many options that may not be supported or will not work. A list of the most common options includes: NAT, IKEv2, VPN monitor, DPD, perfect forward-secrecy, certain timeout values and authentication or encryption algorithms, certificate parameters, and how the vendors handle traffic selectors or proxy IDs.

Monitoring

You have a wide number of ways to accomplish monotoring of the topology, but it all depends on what infrastructures you have to start with or are willing to implement. SNMP is the optimal choice as you can monitor multiple parameters, not just the IPsec tunnel, to figure out if there are any faults or unusual problems. It also enables you to be more proactive.

Monitoring the IPsec tunnel itself is necessary because you can lose connectivity between the gateways and still have an IKE and IPsec SA active, which means that you still keep forwarding packets on that link. This results in a black hole for that traffic, which is why you need a function that can decide when you cannot reach the other gateway (peer). For IKE, you have a function called DPD (dead peer detection) that monitors if the remote end-point is responding and for IPsec you have VPN monitor. If you build route-based IPsec VPNs, you could also use BFD (Bidirectional Forwarding Detection) to monitor the active routing path.

DPD helps you to monitor if the remote end-point is responding. If the remote end-point does not respond to a DPD message, it will tear down IKE and try to re-establish IKE with one of its configured peers. You can configure up to four peers per tunnel for redundancy. If you use IKEv2 you will also tear down the corresponding IPsec SAs.

VPN monitor can monitor if the remote end-point responds to an ICMP request, which could be on the remote network, but it won't verify that the remote IKE peer is working properly. This means it will try to re-establish that SA until the IKE SA times out. Then a full re-establishment of both IKE and IPsec will take place trying to establish a new tunnel.

NOTE There might be interoperability issues with vendors when using VPN monitor.

When using route-based IPsec VPN tunnels, you can also use BFD to monitor the remote neighbor to define if you should converge your routing path.

These three options have different choices on how to optimize the traffic and convergence. Please refer to the software documentation when implementing.

It's highly recommend to monitor functions per path using SNMP, meaning that if you lose the gateway to a remote location, you only have to generate an alarm for that device. You should configure the system to notify the operator that everything behind this point is considered to be "out of operation," if it could not sustain local connectivity for a certain application, or to perform route convergence, meaning that you will get connectivity over another path.

The SNMP server monitors all infrastructure devices on its way to the end target, which means that only the closest device to the SNMP server will trigger and state that the connectivity is down (instead of doing it per function as that will flag multiple alarms). See Figure 1.1. If the tunnel is down, there is no need for the SNMP system to report that all devices behind FW-1 are not accessible. This should be stated clearly in the SNMP instructions. If the router is instead marked as unaccessible, it should only highlight that everything behind that device is lost.

Figure 1.1 SNMP

Security Considerations

Always try to use certificate-based authentication, as preshared keys for authentication are normally weak and not often changed. It's also a management nightmare to manage hundreds or thousands of different keys. Use these general guidelines:

- When choosing a topology, try to avoid IKEv1 using aggressive mode with preshared keys, as that is vulnerable to security injections and can be cracked. If you need to run IKEv1 in aggressive mode, we highly recommend that you use certificate-based authentication.
- When choosing authentication algorithms, try to avoid MD5 and SHA1.
- When choosing encryption algorithms, try to avoid DES and 3DES.
- When choosing Deffie Helman groups, try to avoid groups below 14.

There are three parameters that define when you should change the crypto key. For IKE, you have a *lifetime* that should be considered. With a strong authentication algorithm together with a strong preshared key or certificate, you can use the regular lifetimes. For the IPsec part, you should consider the amount of data that is passed over the link when defining your lifetimes; it can be either in seconds or kilobytes.

Authentication

How you authenticate your device is one of the more important aspects to consider when it comes to security concerns. There are two options available: either a pre-shared key (shared secret), or certificate-based authentication. What is the difference?

*Preshared Key*s: Preshared keys (PSK) are generated by a person or a password generator, which means that anyone else can come up with the same key. With the right knowledge, it's more secure to have this key generated by a person who understands how to construct a secure password versus using a password generator that in most cases uses some kind of dictionary. Keep in mind that most customers that use PSK use the same key on the majority of their gateways; it's very common that this key is not changed or updated for several years. (So what happens if someone else can get their hands on that key or brute force that key? - What run rate of staff and consultants does your organization have?) IKEv1 aggressive mode is weak when matched with PSKs.

Certificates: A certificate is like a passport and should only be issued by a trusted certificate authority (CA). Before you can request a certificate, you first need to generate a unique key-pair to be used and a few parameters unique to the request. This means you can only load this certificate on a specific device and not any other device, which means that this certificate will be unique and can be trusted. When the CA receives the request, the system and/or operator needs to verify that the request is coming from a trusted party that can authenticate itself. When that is done, the system will sign and publish this certificate as trusted. This certificate should then be loaded on the system for authentication to other parties. When this certificate is revoked, any remote peer can reject access with this certificate by verifying that the certificate is revoked and not authentic anymore. This simplifies management of authentication as the CA keeps control over any issued certificate.

Architectures and Topologies

Let's review the different IPsec topologies that offer connectivity between sites and end with a matrix where you can see what functions each topology supports, for example, support for dynamic IPs and different dynamic routing protocols.

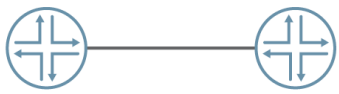


Figure 1.2 *Site-to-Site Topology*

Site-to-Site Topology

A site-to-site IPsec topology (see Figure 1.2) allows two or more locations to securely communicate through the encrypted IPsec tunnel to reach resources on the other side. Finding its way to either end of the tunnel is achieved by IP routing; this can either be static or dynamic depending on implementation. There are a lot of other considerations that you must take into consideration to understand whether this topology works or not. Normally you only use route-based VPNs for this concept, as both end-points can establish the tunnel. You can use both AutoVPN and Auto Discovery VPN too, but then you must be aware that it's always up to the spoke to establish the tunnel.

Hub-and-Spoke Topology

Hub-and-spoke topologies utilize a *so-called* site-to-site topology. This topology is used when a remote site also needs to reach resources behind other remote locations, not just resources behind the central site. When each remote site communicates with another remote site, the traffic path will be via the central site. With traffic flowing through that central hub, you will consume more bandwidth at the hub site as well as require data behind the central site (hub). As you consume more bandwidth, you also introduce higher latency for other remote locations that only need access to the central resources. In the long run, this also impacts the communication between remote sites, as they most likely will be exchanging Voice over Internet Protocol (VoIP) or other latency-critical applications. This then forces the organization to order more bandwidth, impacting the operational expense. Hub-and-spoke VPNs can be achieved with route-based and AutoVPN topologies; if you use AutoVPN, you have to remember that it's up to each spoke to establish the tunnel to the hub.

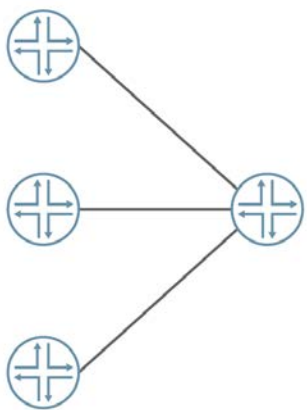


Figure 1.3 Hub-and-Spoke Topologies

Semi-Mesh Topologies

A semi-mesh topology allows traffic to flow between two or more remote locations without utilizing the bandwidth on the central hub. This can be done in two ways. One way is that you build static tunnels between the remote locations that need to communicate. Unfortunately, this requires a lot of man hours to configure and keep up-to-date. What happens, for example, when a new remote location needs to communicate with another one, or one location must be removed? Another issue you'll face with this scenario is that each remote site has a limited number of tunnels. What happens when one remote site runs out of tunnels while tunnels at other sites are not used very heavily? When should the administrator make the decision to remove tunnels from a remote site? Or, should you let the traffic flow through the hub? This introduces a lot of challenges for the organization and a nightmare for the administrator from a daily operation perspective.

The other option is to establish dynamic tunnels between the remote sites at the time when the remote sites need to exchange information. With that functionality, you can remove the problem of a saturated hub and also remove the challenges on how to manage and prioritize which site should not be allowed to saturate the hub. Of course, even in this case, you can run out of tunnel capacity, but that should be part of the planning and sizing of the topology. A semi-mesh topology uses a mix of Auto Discovery VPN and route-based VPNs. When Auto Discovery VPN (ADVPN) is used, it's the spoke that is responsible for establishing the tunnel.

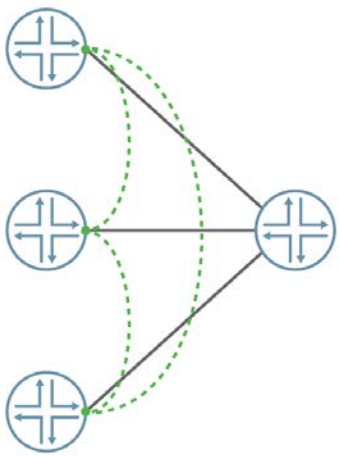


Figure 1.4 *Semi-Mesh Topology*

Full-Mesh Topologies

The full-mesh topology also has two options because it is more detailed than the other mesh-type topologies. The first topology is described as *public* and the second as *private*.

Public Full-Mesh IPsec VPN Topology

This topology uses the same setup as the site-to-site topology, but instead of just configuring a tunnel from one remote site to the central site, you have to build a tunnel between every remote site. This is an enormous job and also introduces an administrative nightmare, which is why it's more common to use a hub-and-spoke or semi-mesh topology. Of course, the full-mesh topology offers the best possible performance and quality of experience for the users or systems that are using this connection. The most important thing you should consider is that each remote site needs to have the ability to set up all possible tunnels; this normally requires a large box, which is more expensive than a smaller one. The other part that really drives up cost is the management of this complex network, because as soon as you need to do a change in the topology, you must touch every device to make configuration changes. The benefit of a public full-mesh that utilizes a traditional site-to-site setup is that it can be used on any IP network versus the private full-mesh setup described next.

Private Full-Mesh IPsec Group VPN Topology (GDOI):

A private full-mesh topology can only run when you either have a private cloud like MPLS, or use all public routable IP addresses where no network address translation will occur. The benefit with GroupVPN is that you use a central key-server that is responsible for the SA that will steer and encrypt traffic. In the site-to-site topology a pair of SAs was used between every connection, whereas in GroupVPN, you only have a single SA that steers all tunnels between the remote sites that is configured in an entity called a *group*. With groups, you can build and define multiple groups that allow traffic to flow only between selected locations. All configuration is done on the key-server and then distributed to remote locations that have an active connection with the key-server. From an administrative point of view this drastically reduces the workload, as well as operational cost.

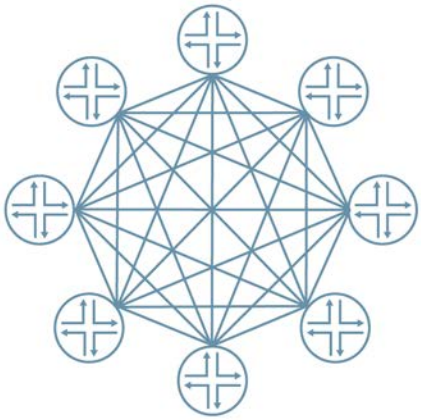


Figure 1.5 *Full-Mesh Topology*

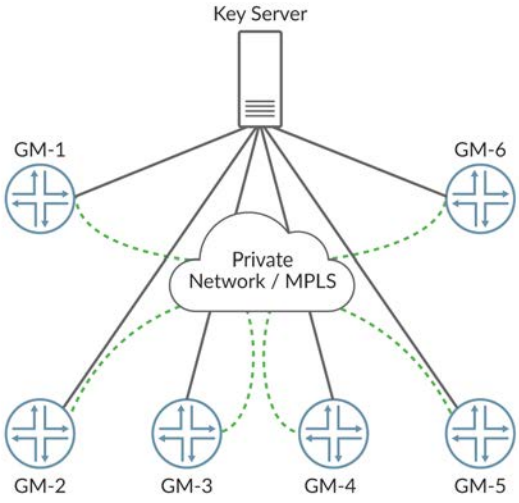


Figure 1.6 *Group VPN*

Topology Matrix

Now let's map these topologies to specific SRX Series VPN features, so the rest of the cookbook can focus in on IPsec VPN concepts and examples. Table 1.1 lists the most common functions customers need to consider when planning their IPsec VPN topologies.

Table 1.1 Topologies Mapped to SRX Series VPN Features

SRX Series Features	Route-based VPN	Auto VPN	ADVPN	Group VPN (GDOI)
Static External IP	Yes	Yes	Yes	Yes
Dynamic External IP	Yes	Yes	Yes	—
IKE Version1	Yes	Yes	—	Yes
IKE Version 2	Yes	Yes	Yes	—
IKE External Interface in VR	Yes	Yes	Yes	—
Dead Peer Detection (DPD)	Yes	Yes	Yes	—
VPN Monitor	Yes	Yes	Yes*	—
Authentication PSK	Yes	—	—	Yes
Authentication Cert	Yes	Yes	Yes	—
NAT Inside Tunnel	Yes	—	—	—
NAT-T for IKE Peer	Yes	Yes	Yes	—
ST Interface in VR	Yes	Yes	Yes	—
Static Routing	Yes	Yes	—	Yes
Dynamic Routing RIP	Yes	—	—	Yes
Dynamic Routing OSPF	Yes	Yes**	Yes	Yes
Dynamic Routing BGP	Yes	Yes**	—	Yes
Auto-Route Insertion (ARI)	—	Yes***	—	—
Hub-and-Spoke	Yes	Yes	Yes	—
Dynamic Spoke-to-Spoke		—	Yes	—
Full-Mesh with a Single SA	—	—	—	Yes
IPv6 Tunnels 4over6	Yes	—	—	—
IPv6 Tunnels 6over4	Yes	—	—	—
IPv6 Tunnels 6over6	Yes	—	—	—
Works with Third-Party	Yes	—	—	Cisco Only

* VPN monitor is not supported if there is a NAT device between the IKE peers.
** Only for P2MP interface.
***Ony for P2P interface.

Chapter 2

Configure PKI, RADIUS, and EAP-TLS

Each local site should have a client you send traffic from if you want to verify that traffic is floating through the system, or else you need to configure local policies to allow the Junos host to send traffic between certain zones.

You can decide if the IP address belonging to the interface should respond to ping and SSH in each security zone. If you don't want this, these statements can be removed. Remember that you need at least one SSH-enabled port to access and manage the box:

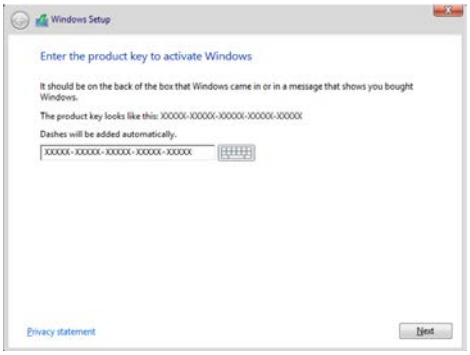
```
set security zone security-zone <zone_name> host-inbound-traffic system-services <function/service>
```

Install Windows Server 2012 R2 Std with Certificate Services for IPsec-based VPNs

Here's the cookbook's first tutorial to help you install a certificate authority (CA) so you can use certificate-based authentication for IPsec.

DISCLAIMER This *Day One* book only shows you how to set up the Microsoft Windows Server 2012 R2 with CA services for a lab environment. It's *not* a best practice guide and this system installation is *not* hardened. For a real, live scenario please consult your Microsoft security expert.

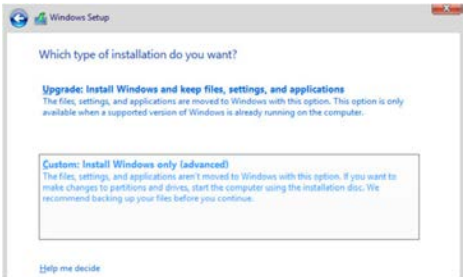
To install Windows Server 2012 R2 Std for Certificate Services follow these steps:



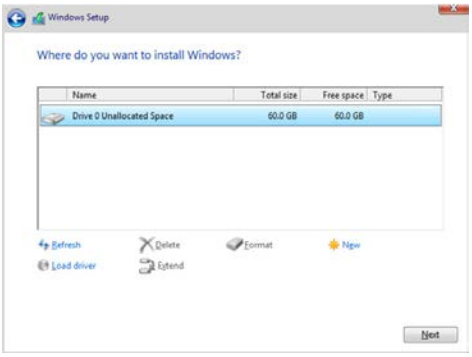
Enter your product license key and click Next.

Choose "Windows Server 2012 R2 Standard (Server with a GUI)" and click Next.

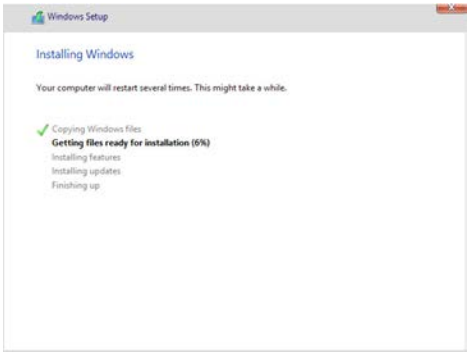
Check "I accept the license terms" and then click Next.



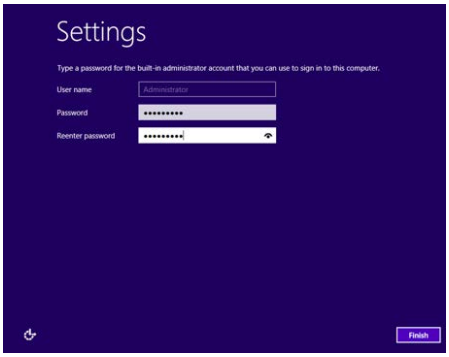
Click on "Custom: Install Windows only (advanced)."



If you have multiple disks, choose the one you want to use. Then click Next.

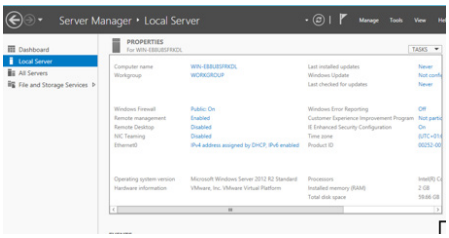


This is just a progress window… or a coffee break.

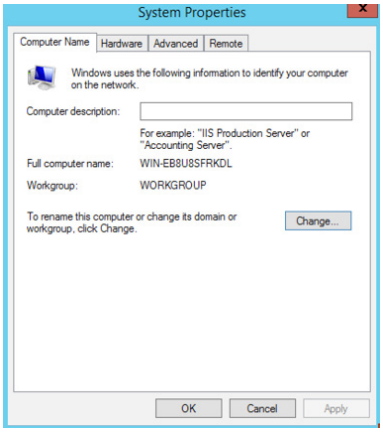


Choose a password for this administrator. Note that it needs to be a STRONG password in order for later services to be installed.

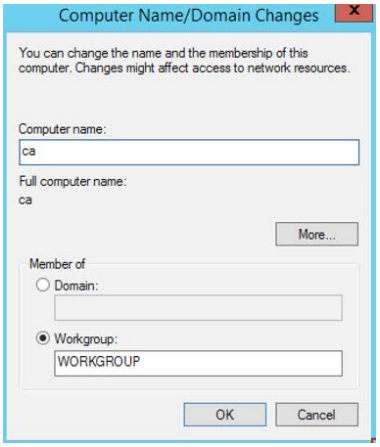
Type the newly-chosen password to log in to Windows.



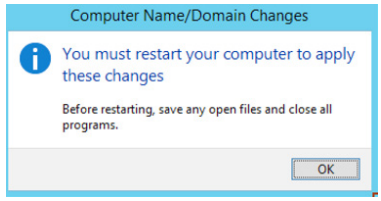
Click “Local Server” on the left-hand side, and then click on the computer name in the properties box.



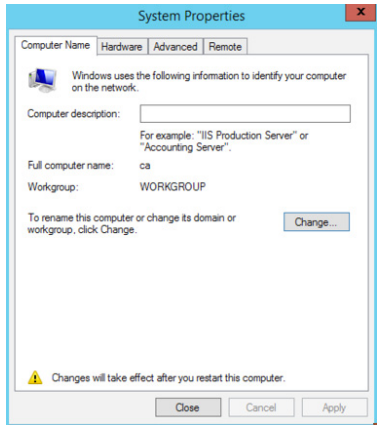
In the “Computer Name” tab, click “Change.”



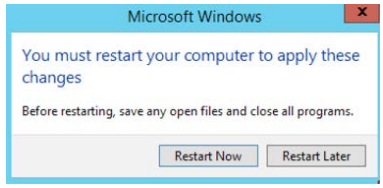
Enter a new name for the server in the “Computer name” box (for example, “ca”) then click OK. By changing the name of the server, you are forced to reboot the server.



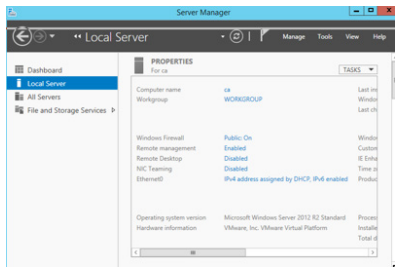
Click OK.



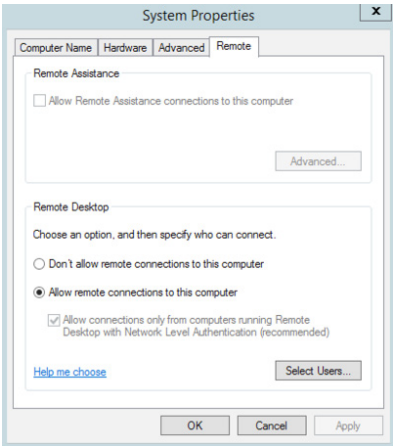
Click Close.



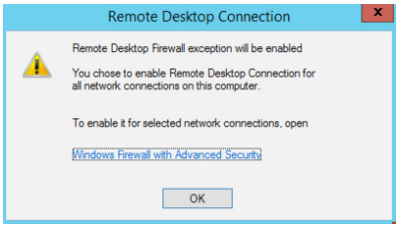
Click Restart Now. When the server comes back up, log in again.



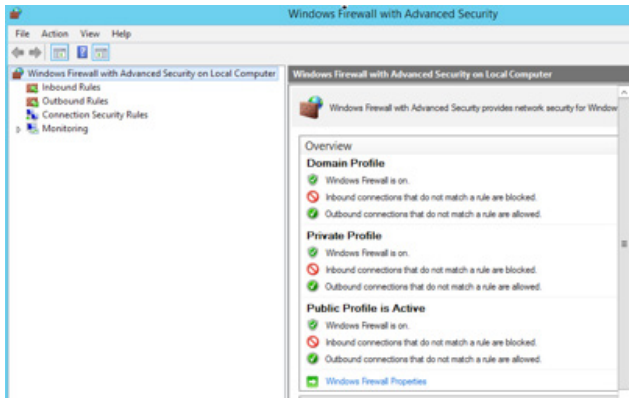
Click “Local Server” and click on the “Disabled” statement next to “Remote Desktop.”



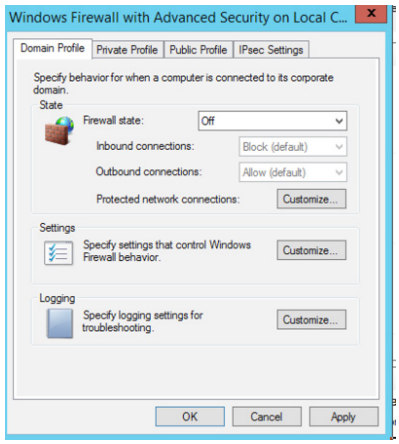
Click on the “Remote” tab then choose “Allow remote connections to this computer.” Click OK.



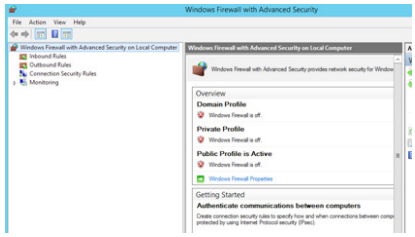
DO NOT click OK! Instead, click on the link “Windows Firewall with Advanced Security.”



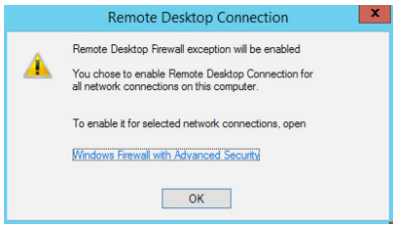
Right-click "Windows Firewall with Advanced Security on Local Computer," then select "Properties."



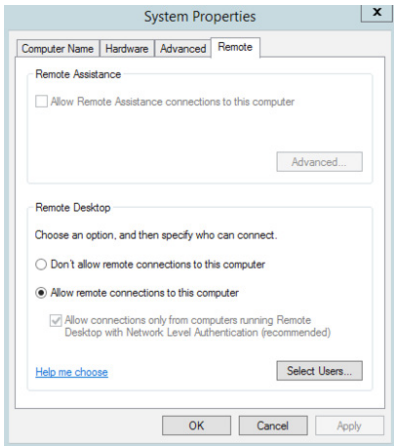
Under the tabs "Domain Profile," "Private Profile," and "Public Profile," set "Firewall state:" to "Off," then click OK.



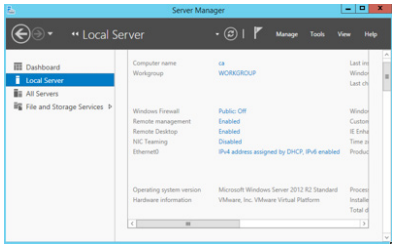
You can now close the Windows Firewall windows by clicking OK on each one until you have closed them all.



Click OK.

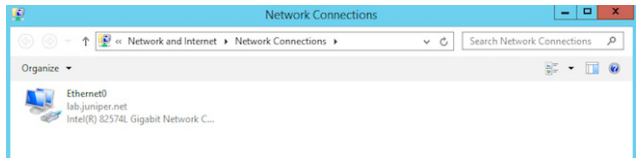


Click OK.

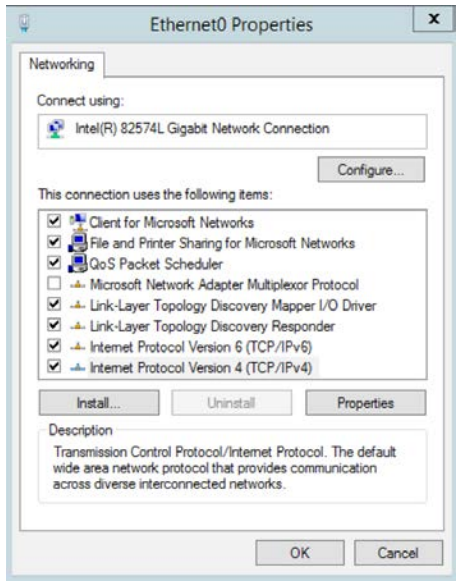


Your view should have a Remote Desktop enabled. If it's not updated, click "Dashboard > Local Server," again, to verify.

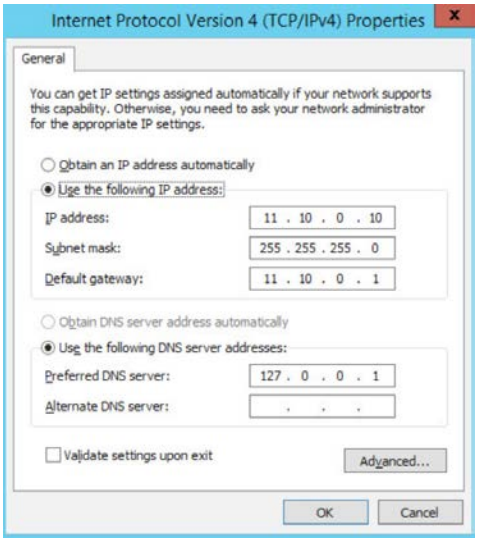
Last of all, you should change the IP-address to be static. Click on the value for Ethernet0: "IPv4 address assigned by DHCP, IPv6 enabled."



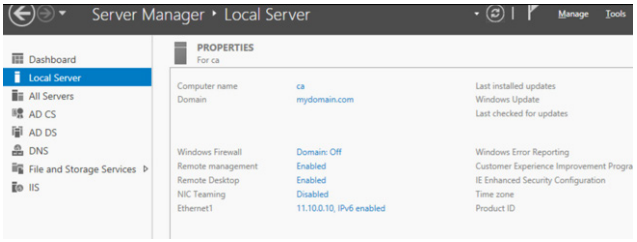
Right-click “Ethernet0” to choose its properties.



Double-click “Internet Protocol Version 4 (TCP/IPv4)” at the bottom of the list.

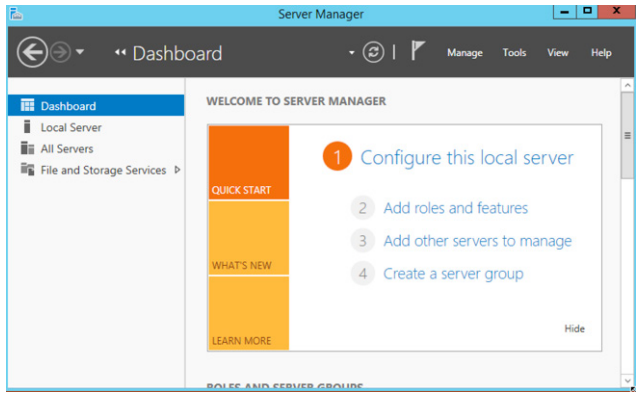


In the "General" tab, enter the IP and DNS information received from your network administrator and then click OK.

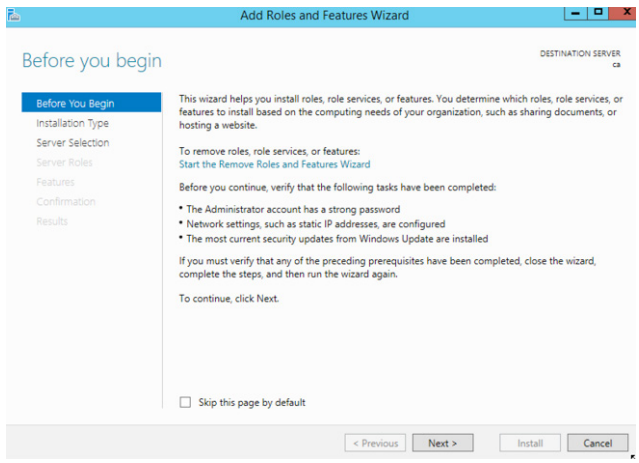


To have complete functionality, you either have to join this server to an Active Directory forest or build a new one. For our purposes, let's build a new one for our lab. Before you do this, however, you should verify that your server has the correct timing/clock and decide if you want to enable Windows Update or not (which is recommended to enable).

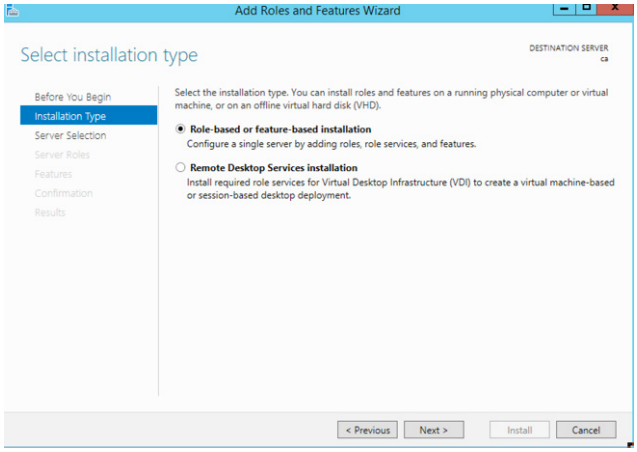
From the Dashhoard…



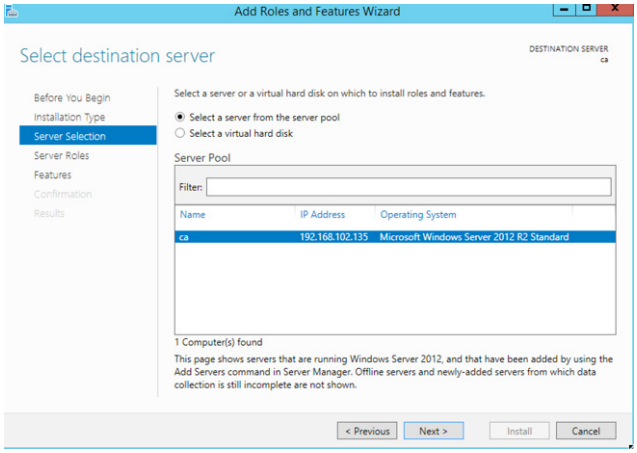
First highlight "Dashboard" then click item 2, "Add roles and features" in the right-hand Server Manager welcome screen.



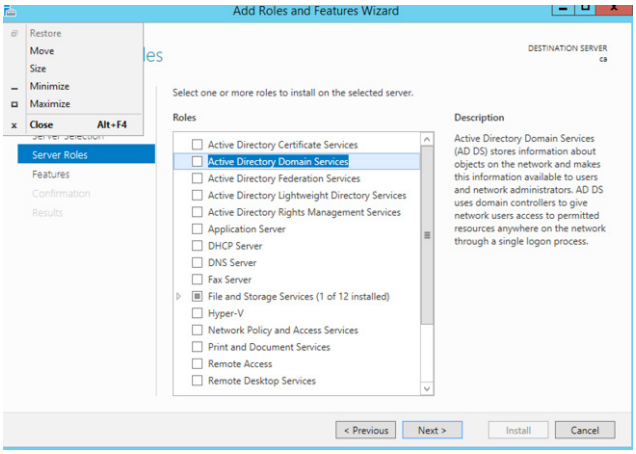
This window informs you that you might have missing dependencies, which should be corrected before you begin. It's a default window, so if the former instruction has been accomplished, you can just click Next.



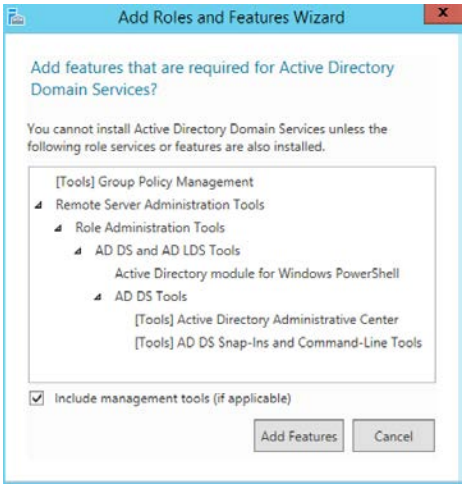
Select the "Installation Type" option for "Role-based or feature-based installation" and click Next.



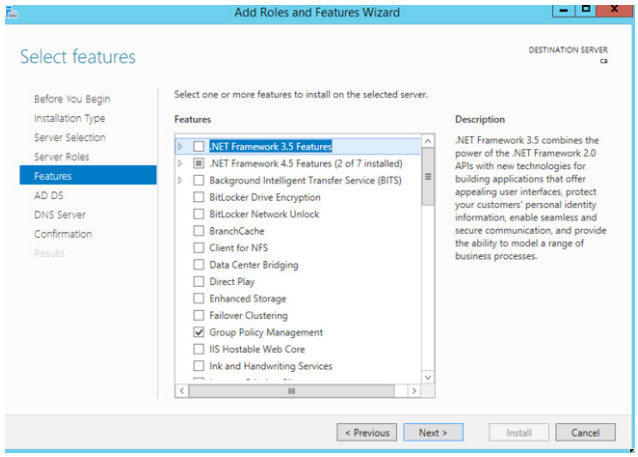
In the "Server Selection" options, highlight "Select a server from the server pool," and then select the server you just installed from the list and click Next.



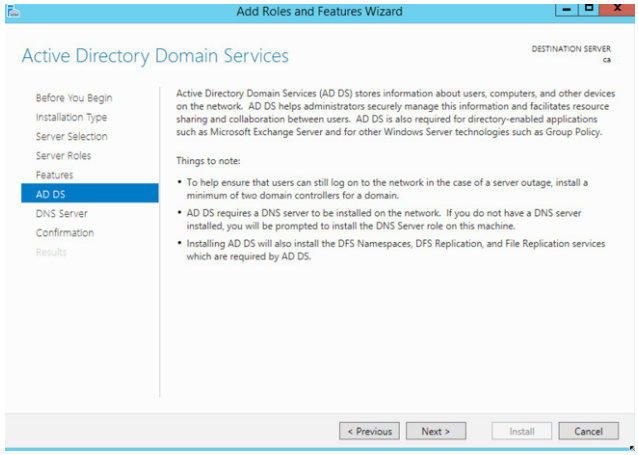
Check the option to install "Active Directory Domain Services" and click Next.



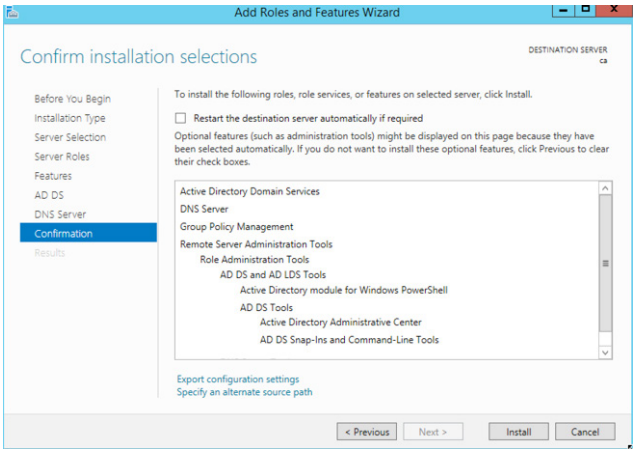
A new wizard begins. Click "Add Features" and then click Next on the following window.



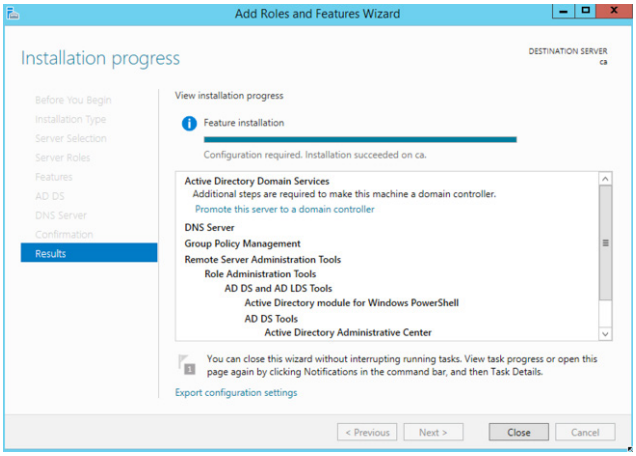
Click Next.



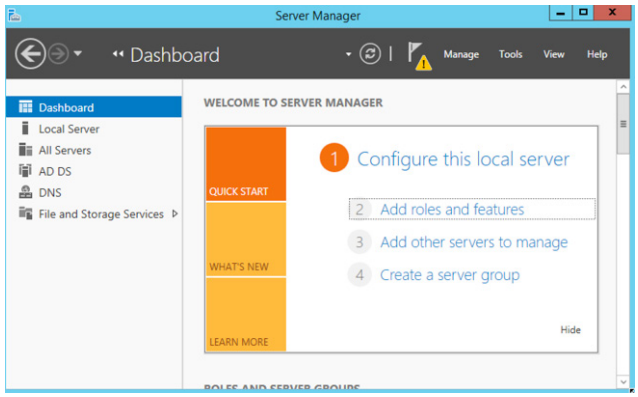
Click Next



Now click "Install" at the Confirm installation selections window.

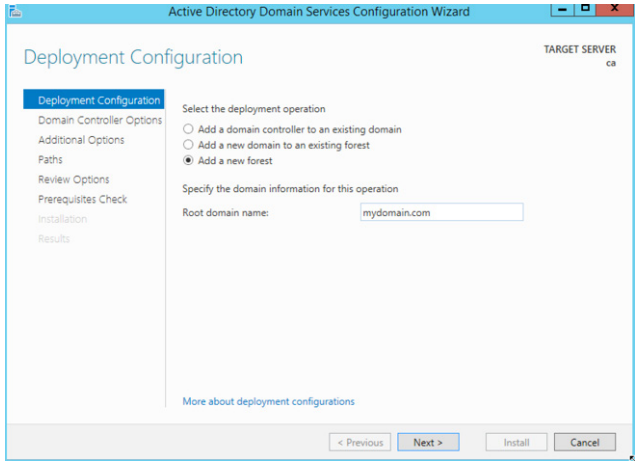


When the installation is done, the progress bar is all blue and "Configuration required. Installation succeeded on xxx" is displayed. Then you can click Close.

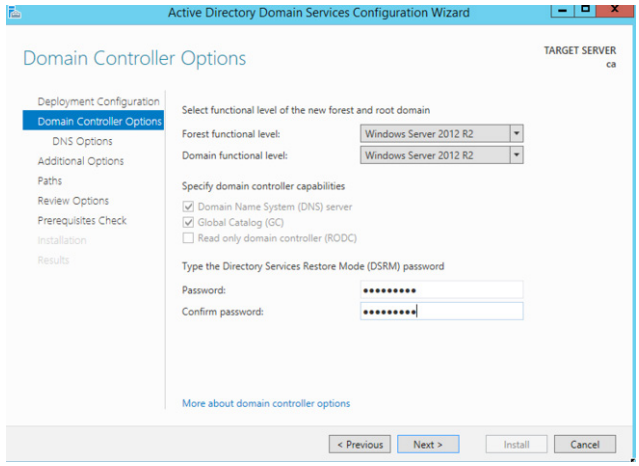


Back to the Dashboard. A yellow triangle in the upper menu bar notifies us to promote this "server as a domain controller." Click on the triangle and then on "Promote this server to a domain controller."

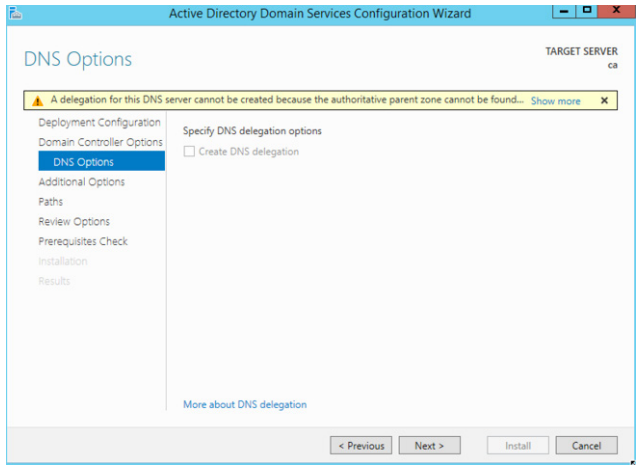
The Domain Services Configuration Wizard launches.



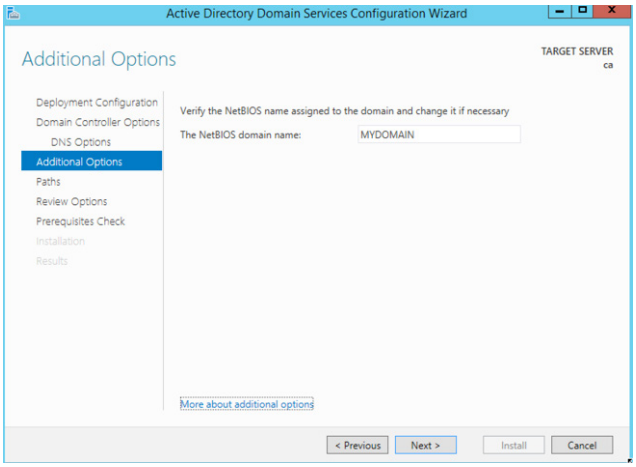
In the first step of the wizard, highlight "Add a new forest" and add your domain name. When finished, click Next.



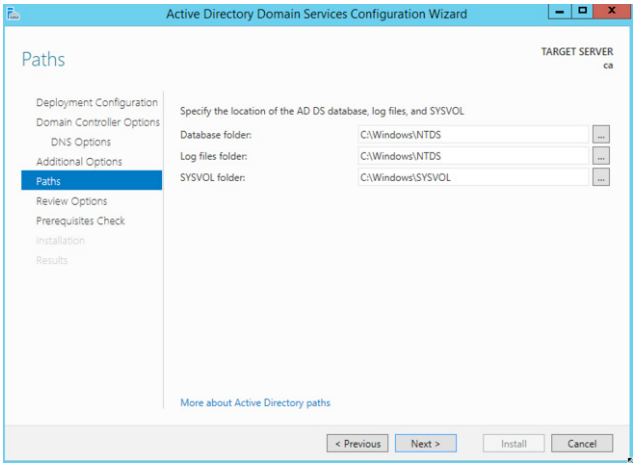
Add a password for "Directory Services Restore Mode (DSRM)" and click Next.



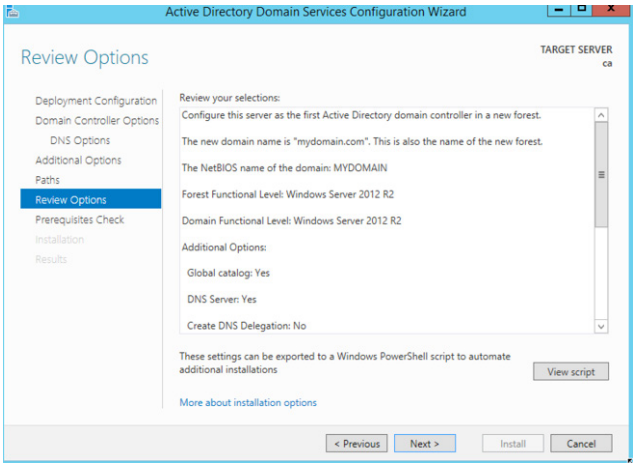
Click Next.



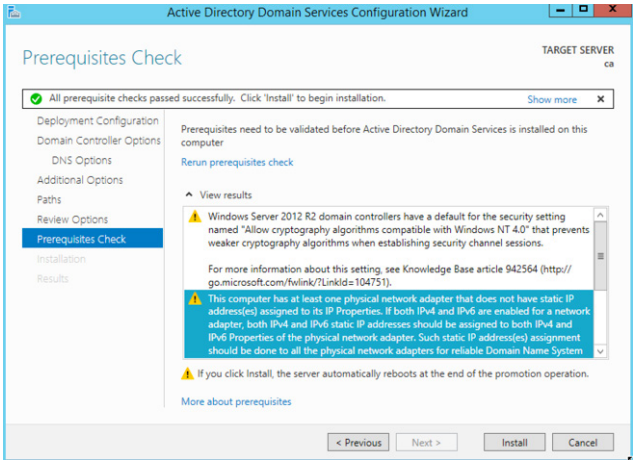
Click Next.



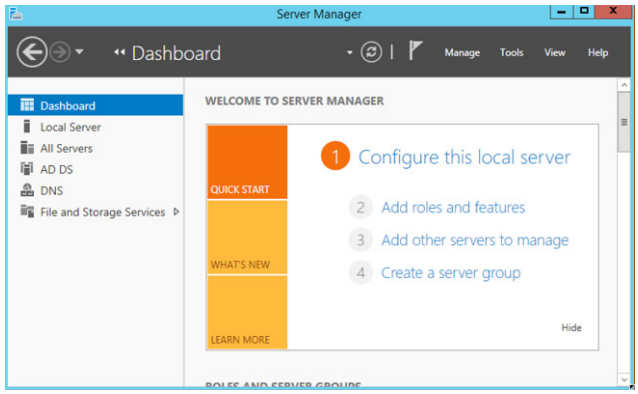
Click Next.



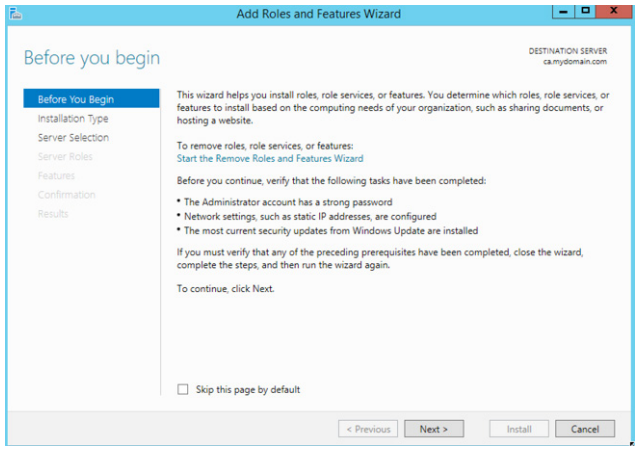
Click Next. It's a fast wizard.



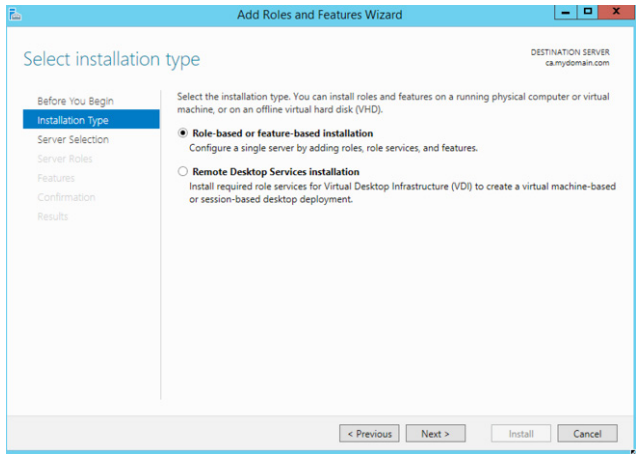
Click Install. When installation is complete, the server will reboot. When the server is up again, log in to the server.



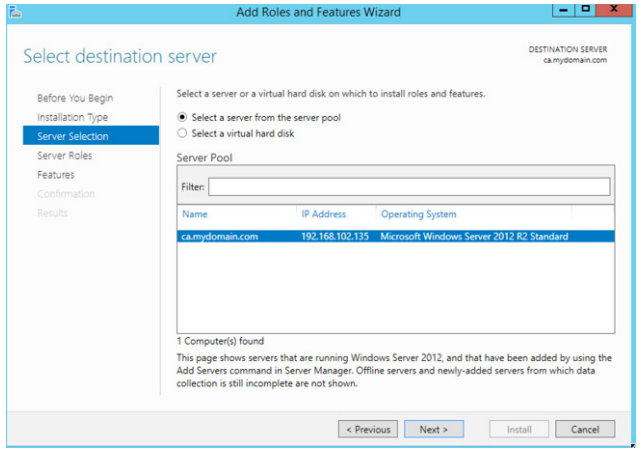
It's now time to add and configure the CA services. Click number 2, "Add roles and features" in the list of options.



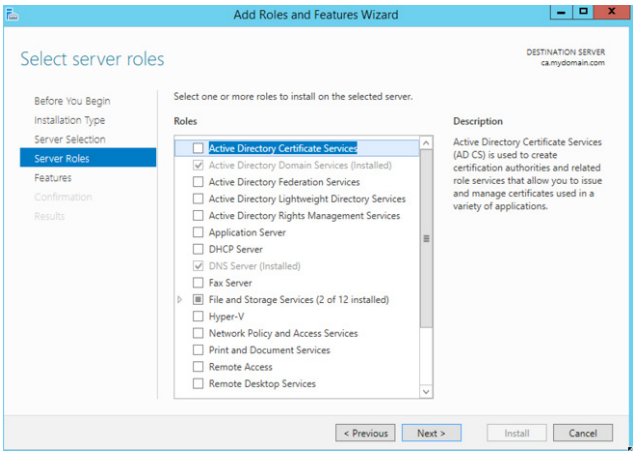
The Add Roles and Features Wizard launches. Click Next.



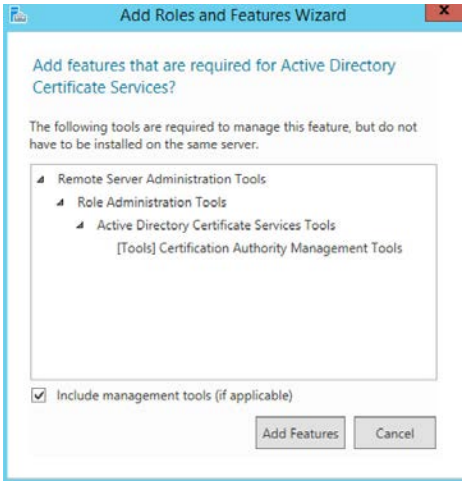
Highlight "Role-based or feature-based installation" and click Next.



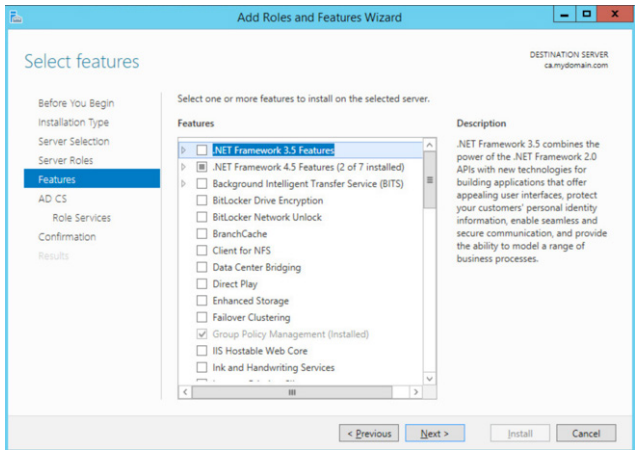
Highlight "Select a server from the server pool" and highlight your server, then click Next.



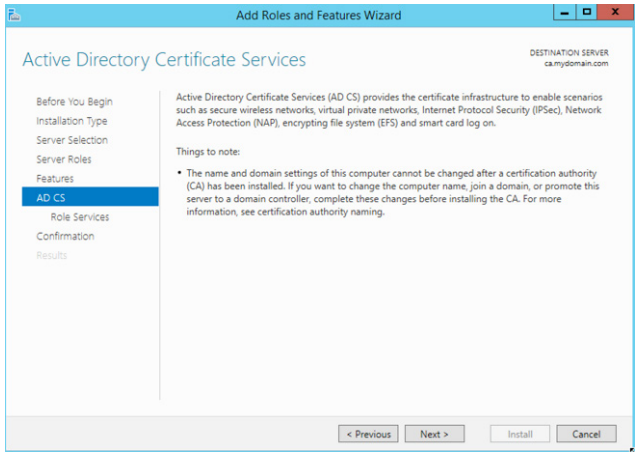
Select "Active Directory Certificate Services" and click Next.



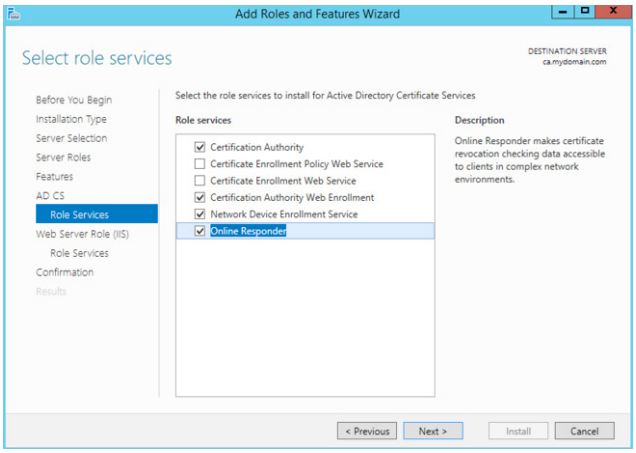
Click Add Features.



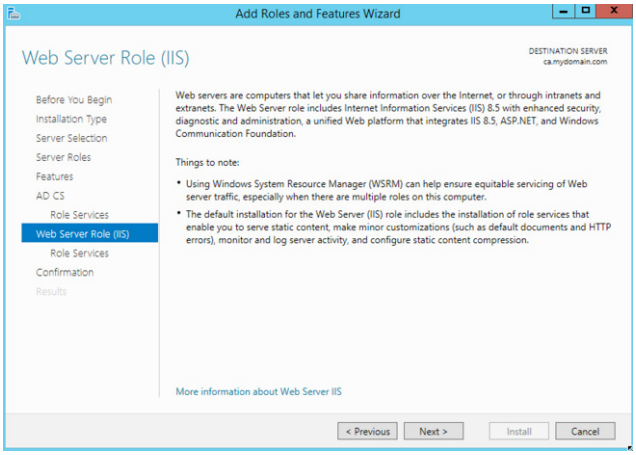
Click Next.



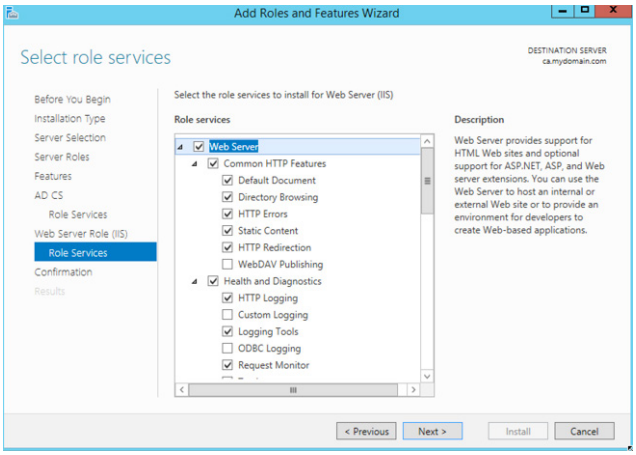
Click Next.



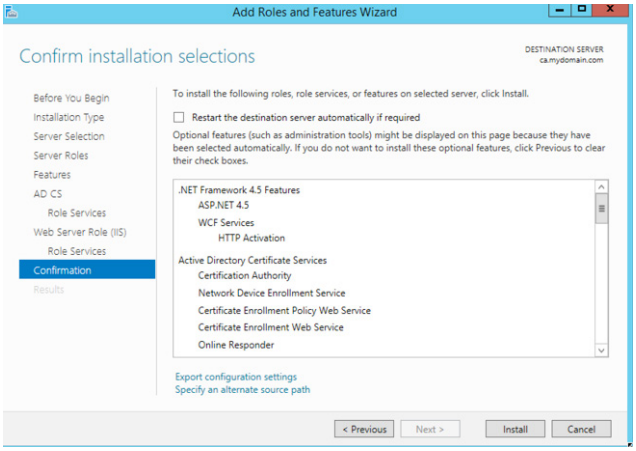
In the Role Services options, check the boxes as shown. For each selection, click "Add Features" in the new window. After each option is checked, click Next.



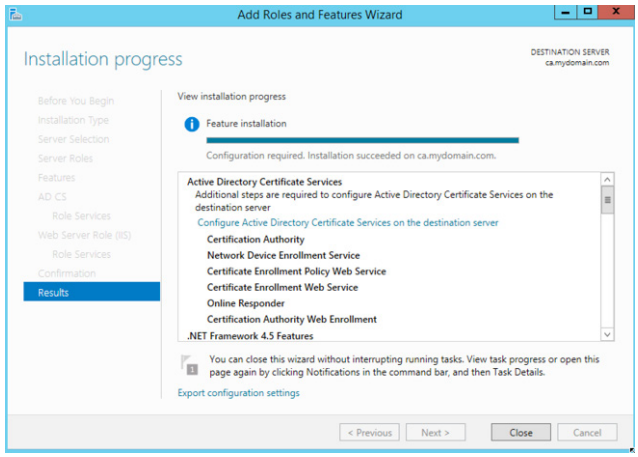
Click Next.



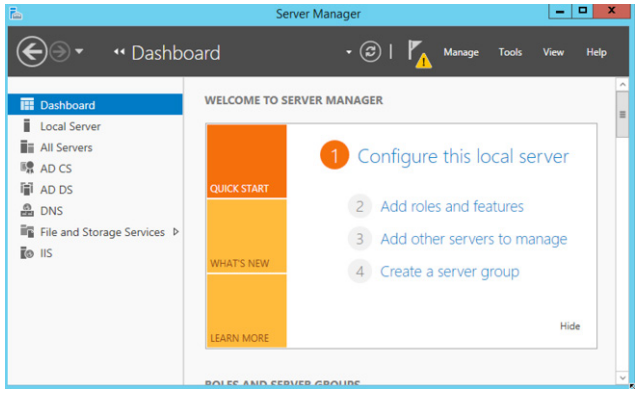
Click Next.



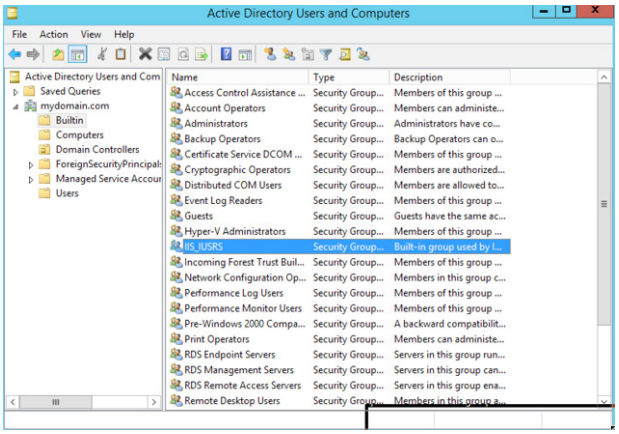
Click Install.



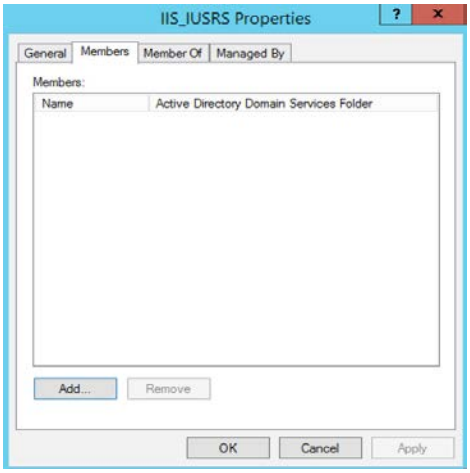
After installation is complete, click Close.



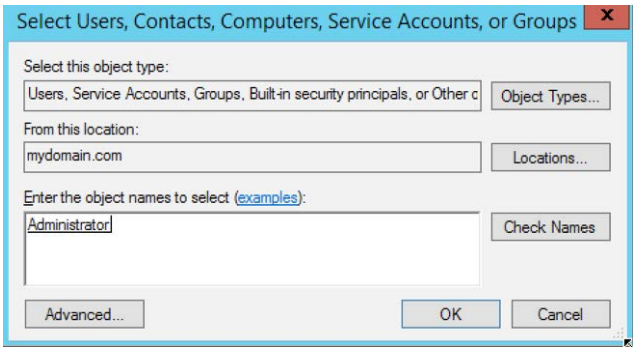
Now that the "Active Directory Certificate Services" is installed you can configure the Certificate Services. Click the "Tools" menu in the right corner of the menu bar, and choose the "Active Directory Users and Computers" option.



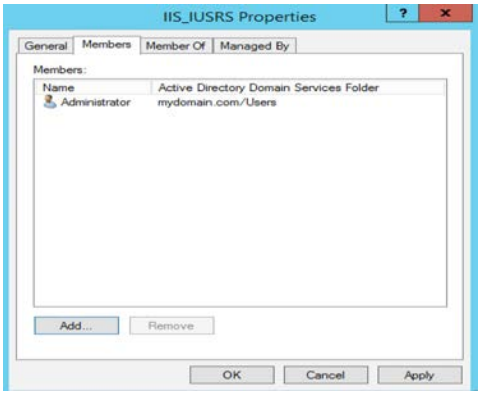
Choose "mydomain.com > Builtin" in the left sidebar and then right-click the "IIS_IUSRS" and choose "Properties" in the pop-up menu, because you need to add a user to this account for the services to work.



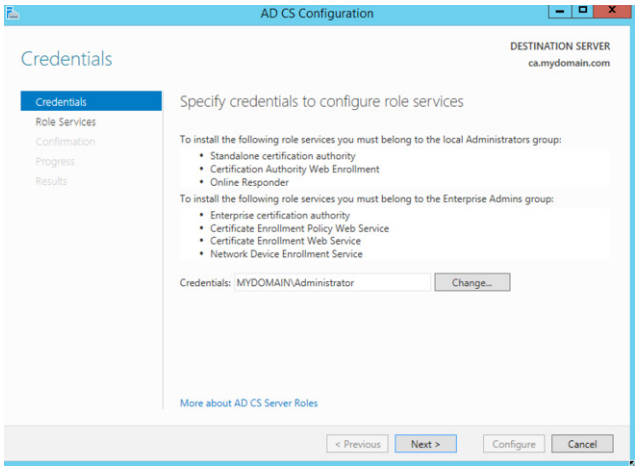
Select the "Members" tab and then click "Add."



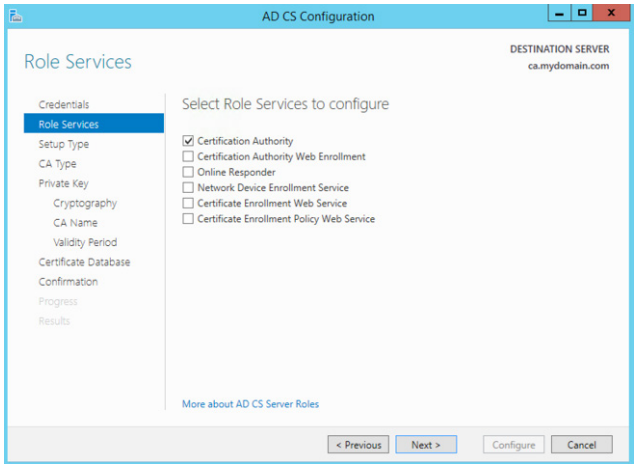
Enter the name of a user with administrative permission (here, "Administrator"), then click the "Check Names" button. When you see that the username is underlined, you have successfully added the user. Click OK.



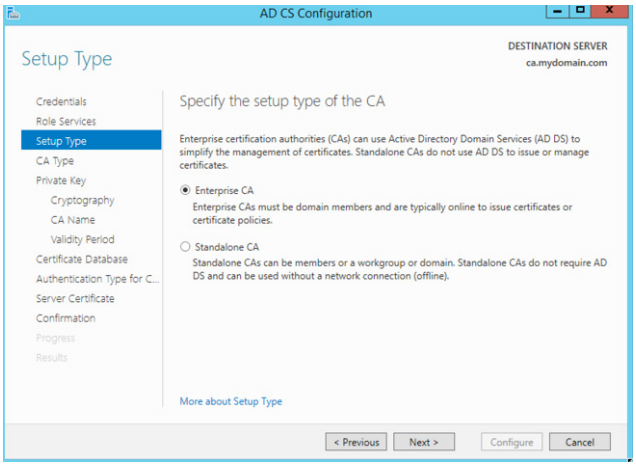
Click OK, and close all the rest of the windows until you return to the Server Manager window. Click the yellow triangle in the menu bar, and click "Configure Active Directory Certificate Services." The Active Directory Certificate Services (ADCS) configuration wizard launches.



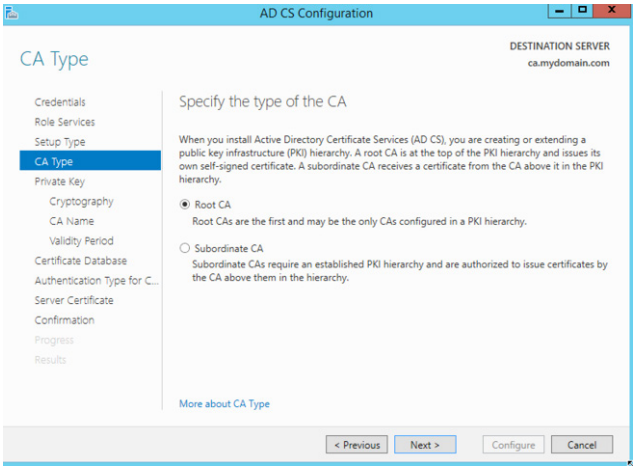
Click Next.



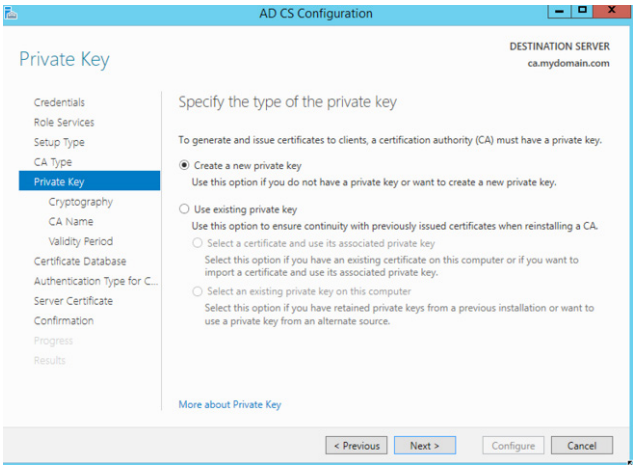
Highlight the service shown and click Next.



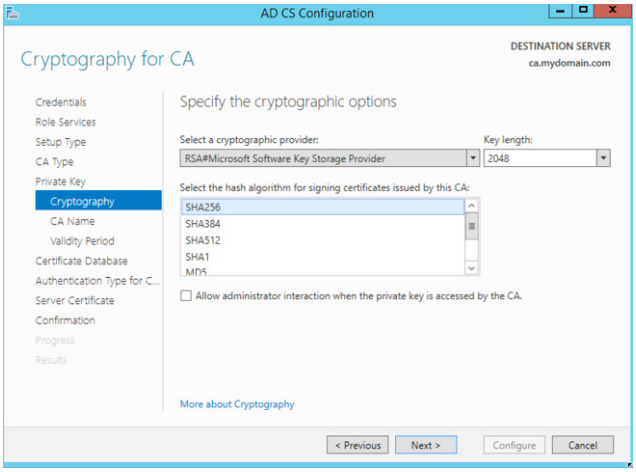
Check “Enterprise CA” and click Next.



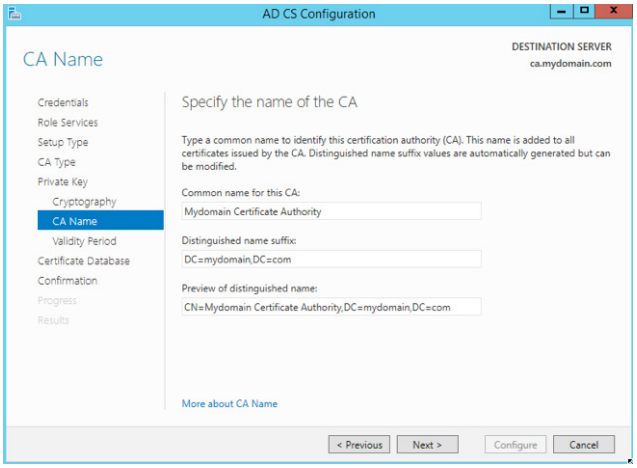
Check “Root CA” and click Next.



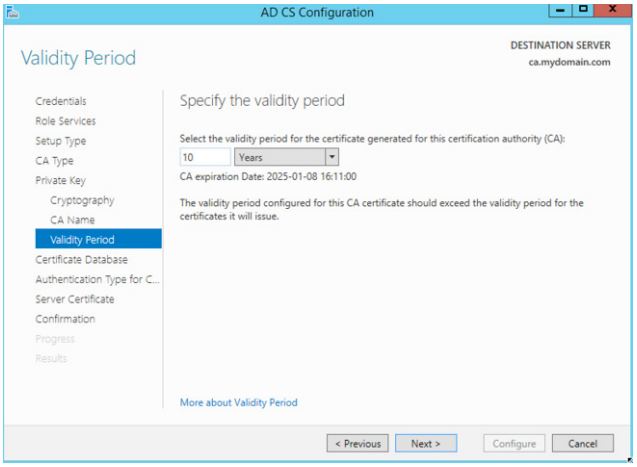
Choose "Create a new private key" and click Next.



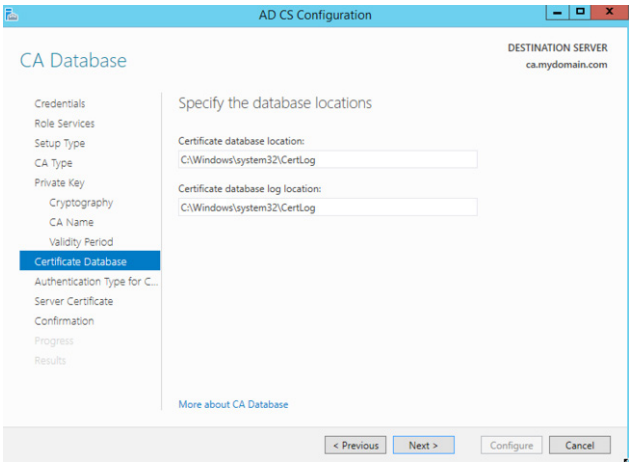
Choose the algorithm shown (here SHA256) and click Next.



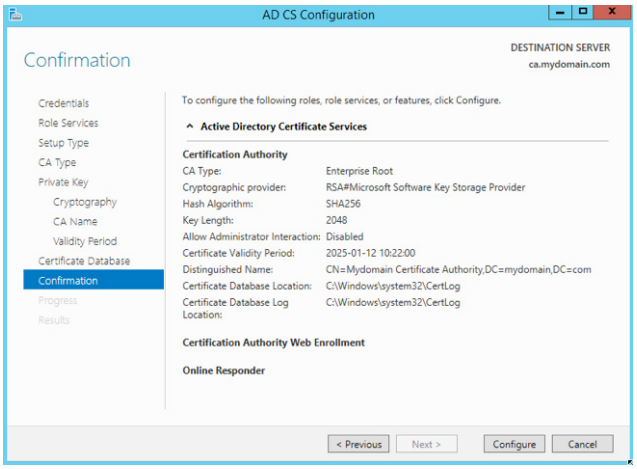
Choose a common name for your host. We chose *Mydomain Certificate Authority*. When entered, click Next.



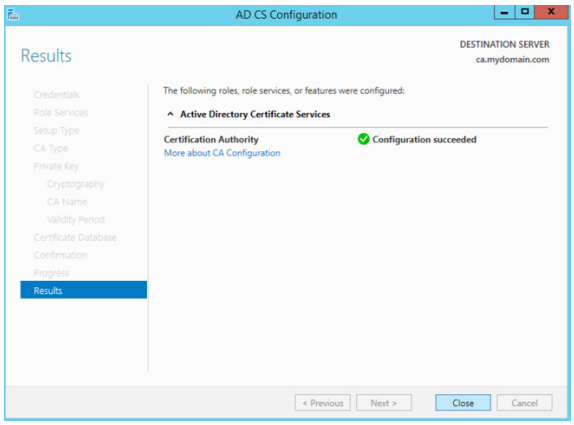
Choose the validity period for issued certificates. We chose 10 years for this lab. When finished, click Next.



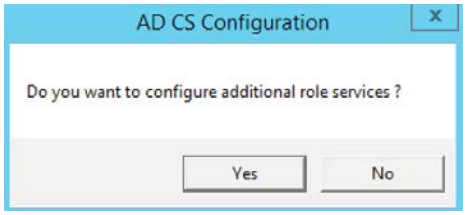
Click Next



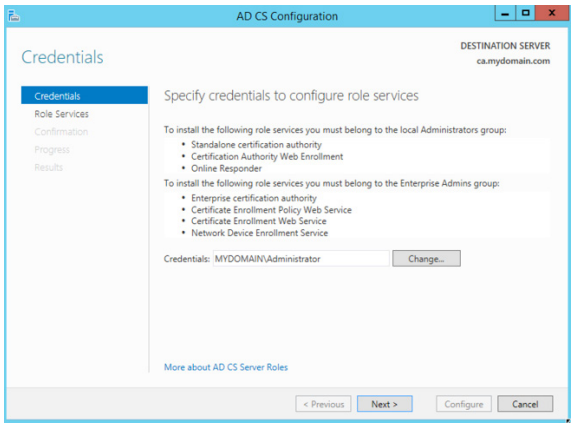
Check your settings and click Configure.



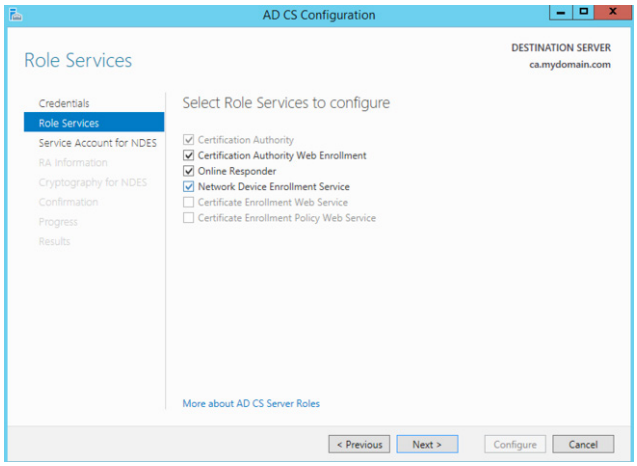
When the three services are installed you'll get a green checkmark. Click Close.



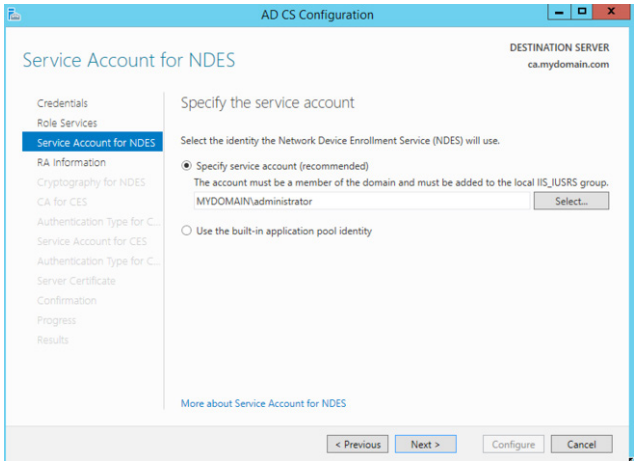
You'll get an opportunity to add more services. Click "Yes" to install the following services.



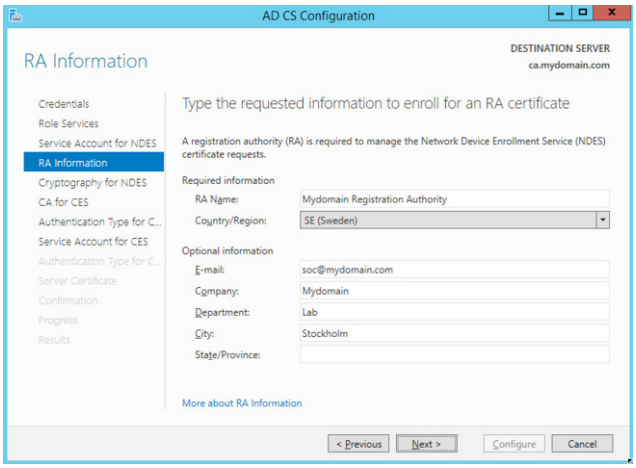
Click Next.



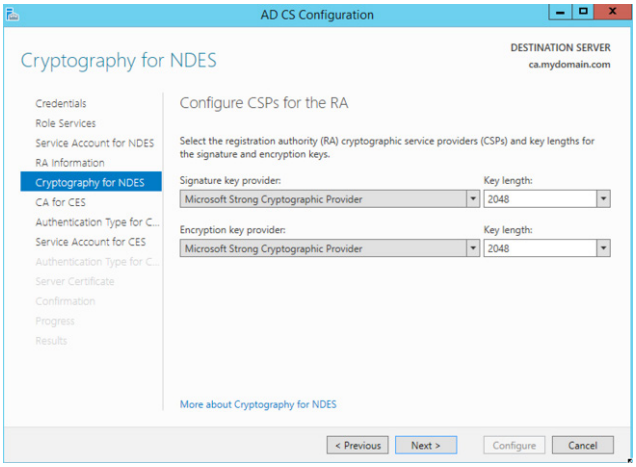
Add the three services shown and click Next.



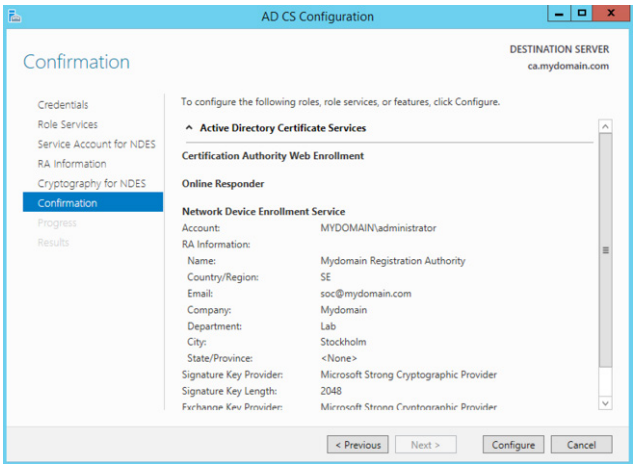
Click "Select" and select the account you added to the IIS_IUSRS account earlier. Then click Next.



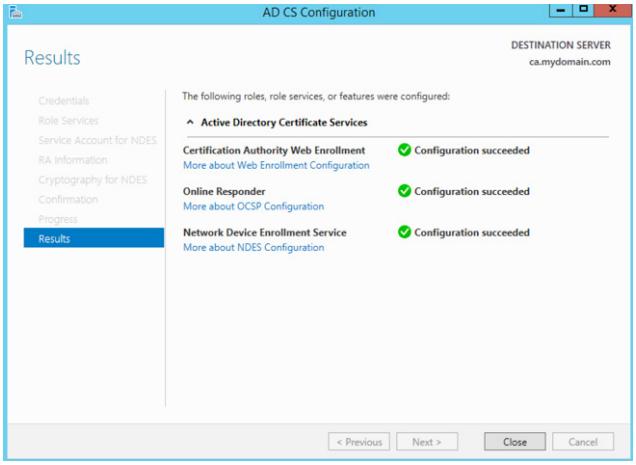
Fill in the form with the information that is correct for your organization and click Next.



Highlight the fields as shown here, then click Next.



Verify your input and then click Configure.

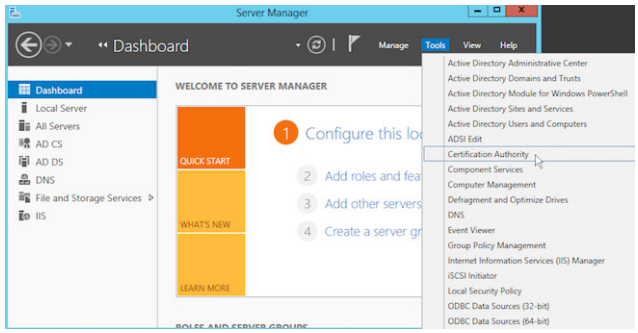


Your Active Directory Certificate Services are installed and ready to use. Click Close.

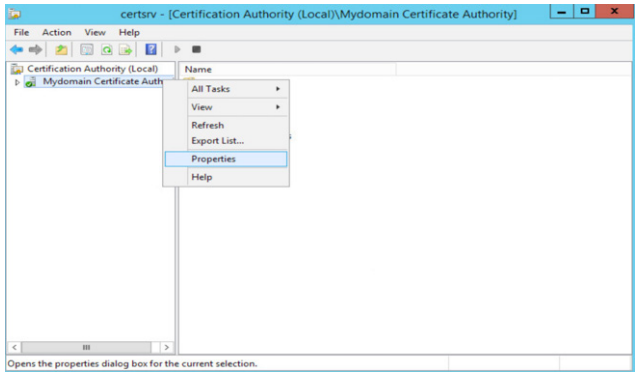
Now you need to configure OCSP for your devices to verify the revocation status of issued certificates from this Certificate Authority. You should also configure SCEP to have a single password (ticket) for each device enrollment.

CAUTION Keep in mind that anyone who has this key and access to the SCEP's URL can later enroll a certificate without any administrative approval.

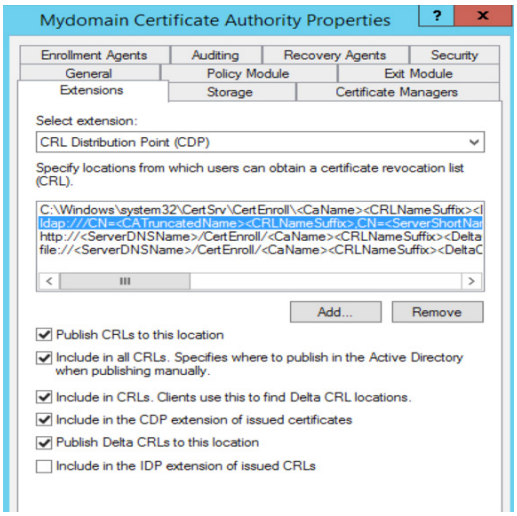
Okay, let's enter the administrative mode of ADCS.



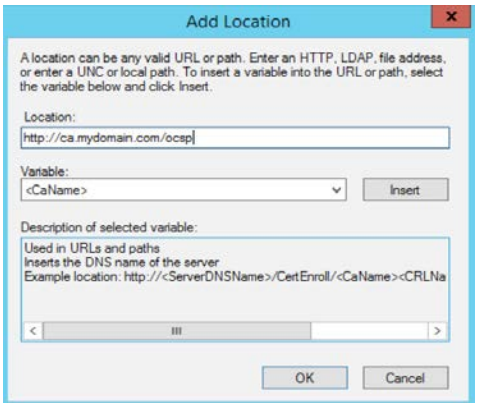
Click on "Tools > Certificate Authority."



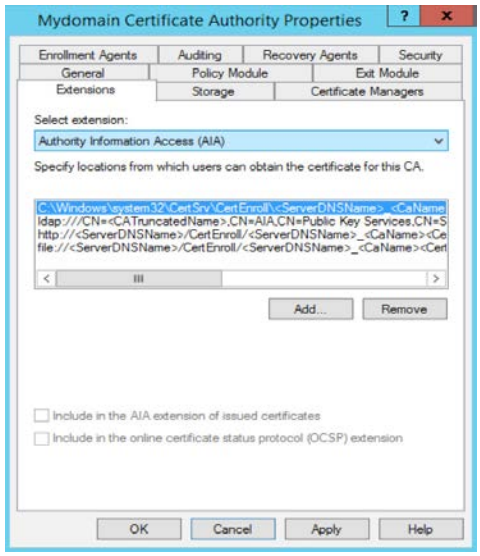
Now, change the OCSP path in the certificate by right-clicking on "Mydomain Certificate Authority" and then clicking on "Properties" in the pop-up menu.



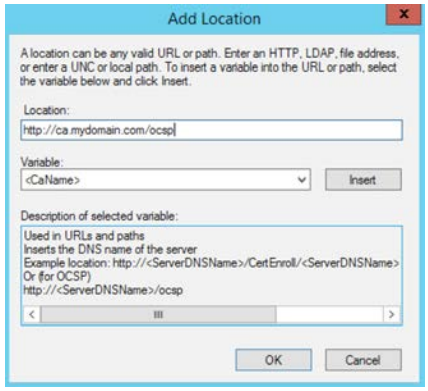
There's a lot here. Under the Extensions tab, remove the three statements for LDAP, HTTP, and FILE, and then click "Add."



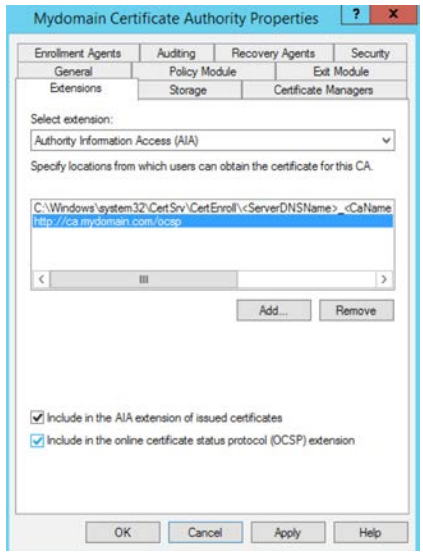
In the Location field, type in a URL that is accessible from all of your machines, as this is the URL each firewall will use to verify if the certificate is still valid. Then click OK.



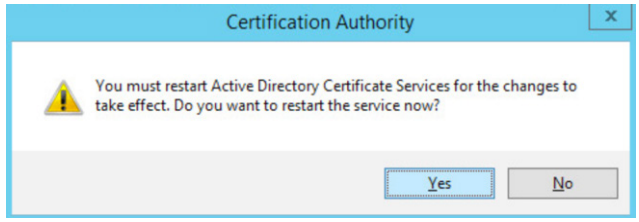
Now you should change the "Select extension" field to "Authority Information Access (AIA)" and also remove the three fields for LDAP, HTTP, and FILE, and then click "Add."



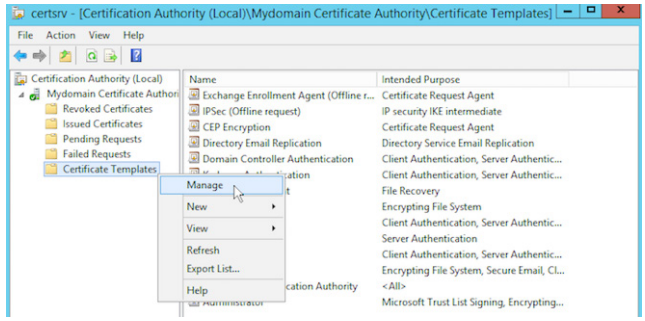
Once more add a URL that is accessible from your devices. Then click OK.



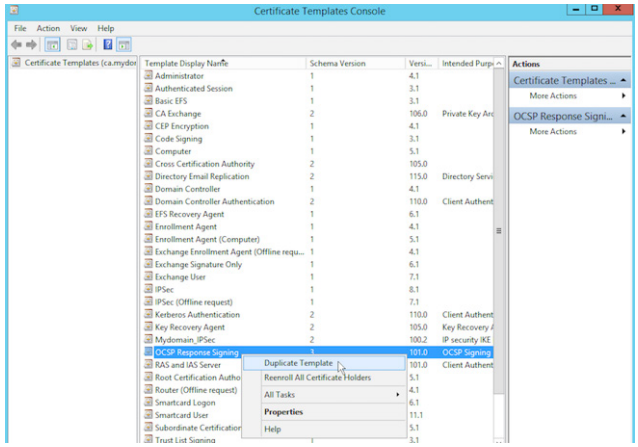
Now check off the two options as shown, and then click OK.



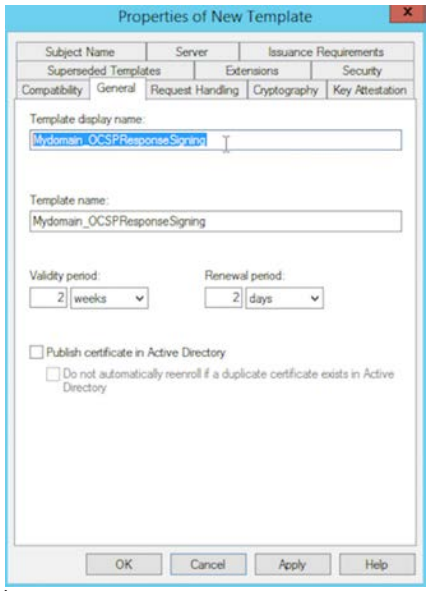
Click Yes. When the service has restarted, go to the next page.



Right-click on “Certificate Templates” then choose “Manage.”

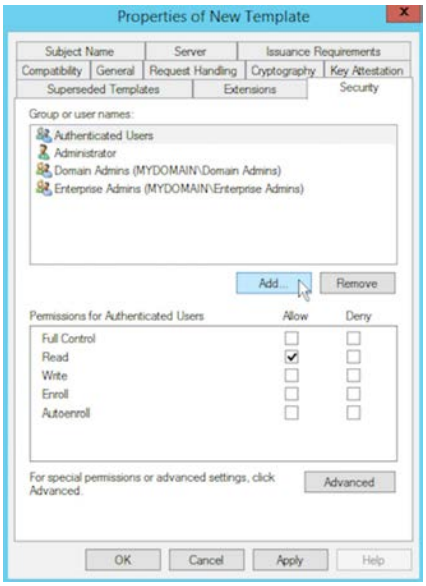


Right-click on "OCSP Response Signing" and choose "Duplicate Template."

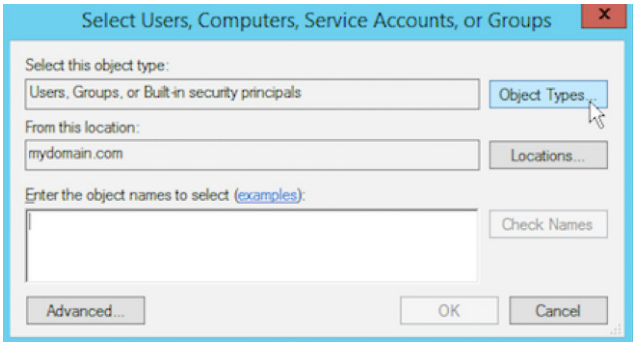


\

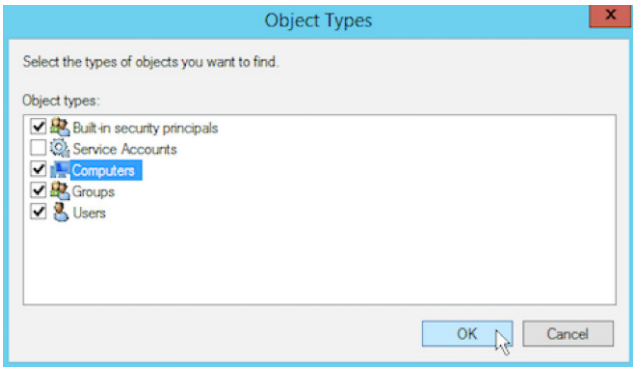
Change the name under the "General" tab. This book uses the name "Mydomain_OCSPResponseSigning." Then click on the "Security" tab.



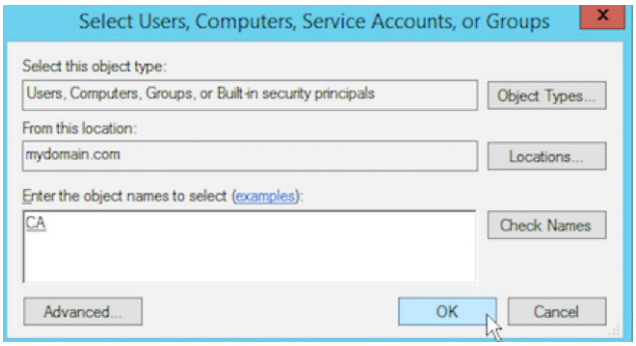
Change the security permissions under the "Security" tab. Click "Add."



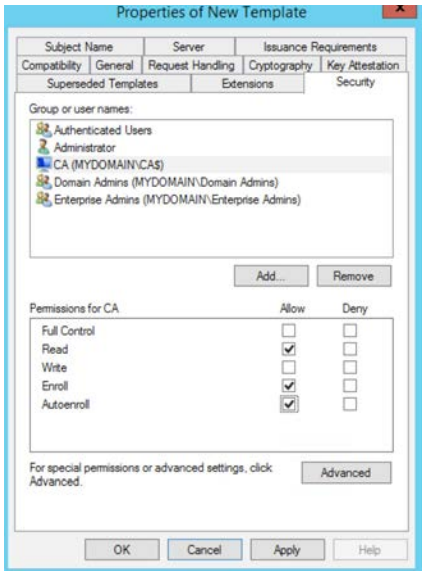
Then click "Object Types."



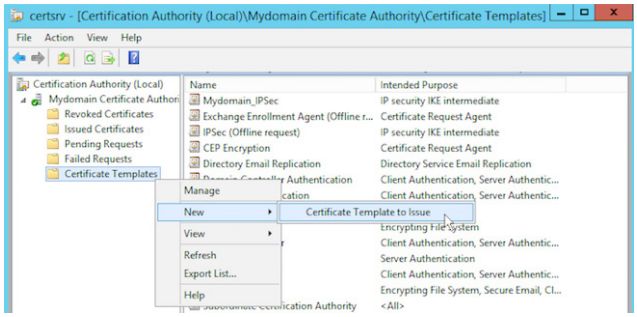
Select "Computers" and click OK.



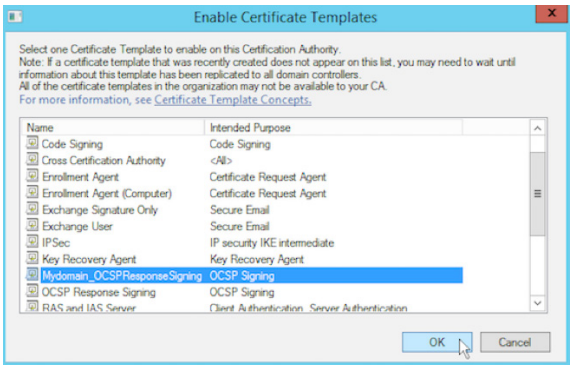
Now you can search for a computer instead of just users. Type the name of your server where "AD CS" is installed ("CA" in our case) and click "Check Names." When you see your computer name with an underline, click OK.



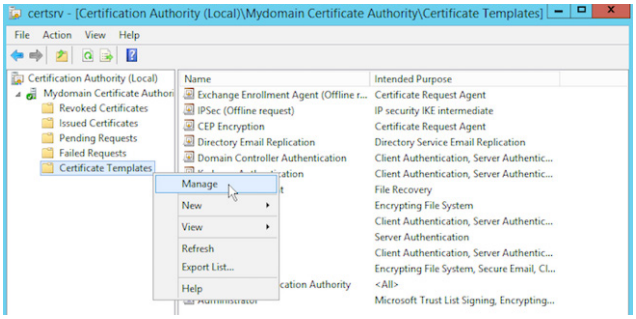
Check the security permissions options for this machine to "Read," "Enroll," and "Autoenroll," then click OK.



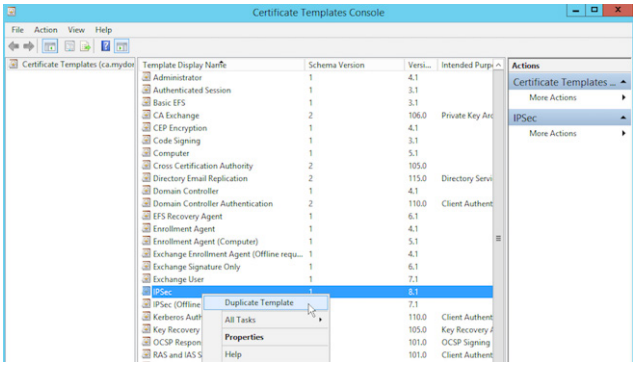
Add this new template to active templates by right-clicking "Certificate Templates" under "AD CS" management.



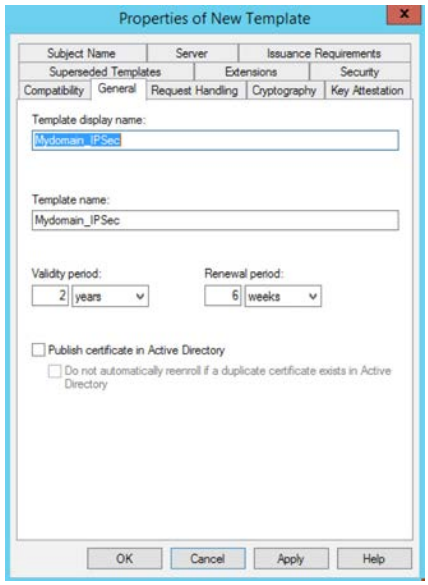
Choose your newly-created template and click OK. And then exit the Template window.



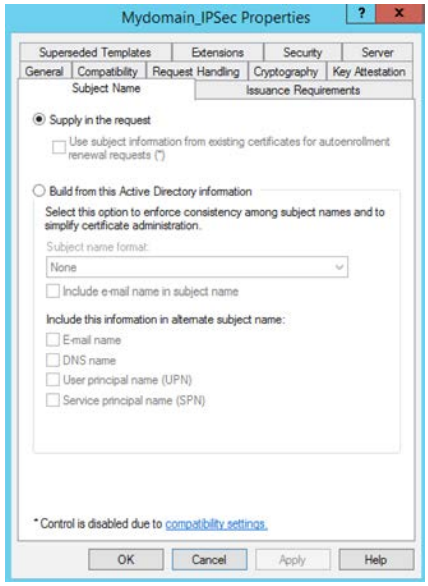
Navigate back to the Certification Authority console and right-click on “Certificate Templates” and select “Manage” in the pop-up menu.



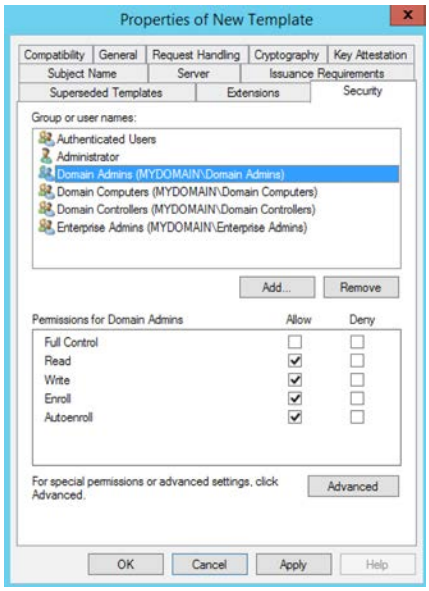
Right-click on the template “IPsec” and select “Duplicate Template.”



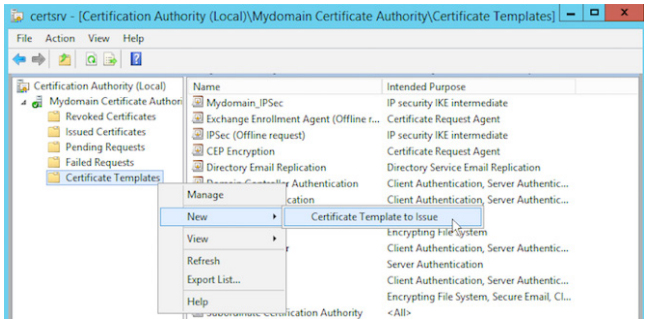
Change the name of this new template under the "General" tab to something that works for you (here *Mydomain_IPsec*). Then click on the "Subject Name" tab.



Check "Supply in the request" and click OK on the window that opens up. Then click on the "Security" tab.

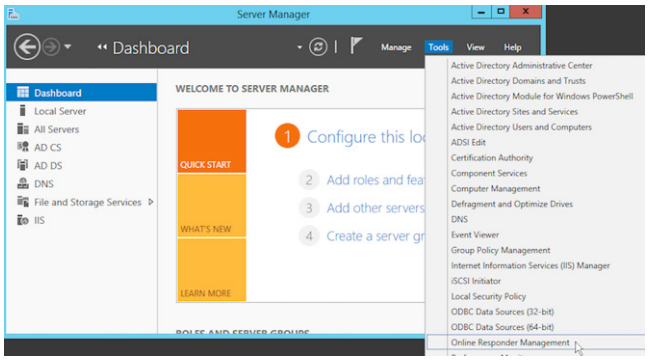


Here you need to change the security permissions under the "Security" tab. Change the "Domain Admins" options to those shown, and then click OK.

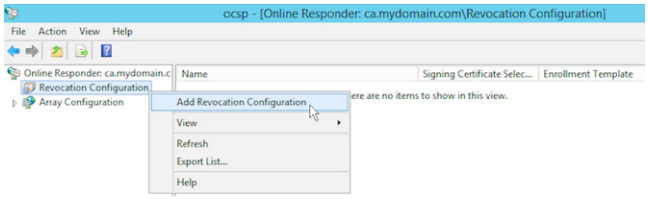


Add this new template to active templates by right-clicking "Certificate Templates" under "AD CS" management. Choose your newly-created template and click OK. And then exit the Template window.

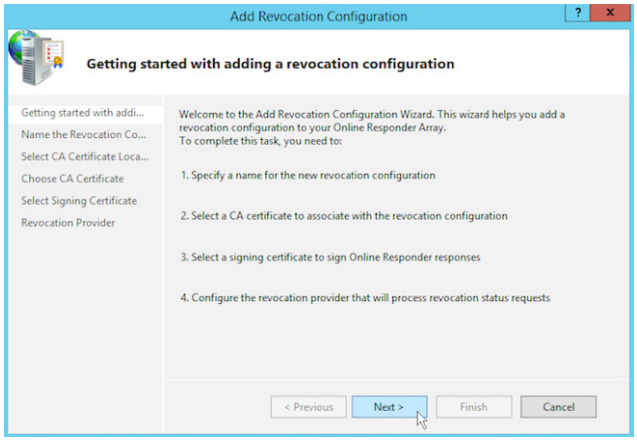
Back to the Dashboard.



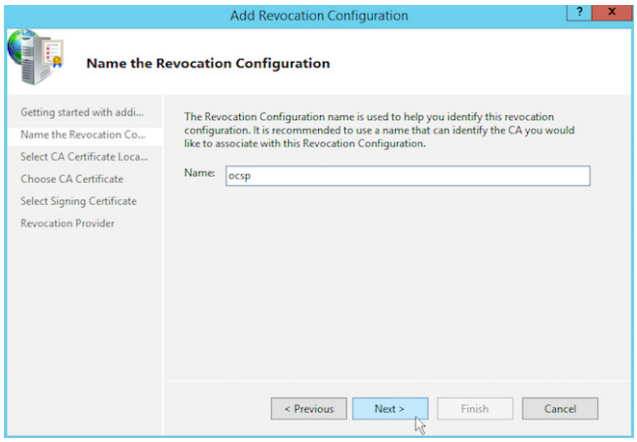
Now let's configure the Online Responder Management module. Click on the "Tools" menu and then the "Online Responder Management" option.



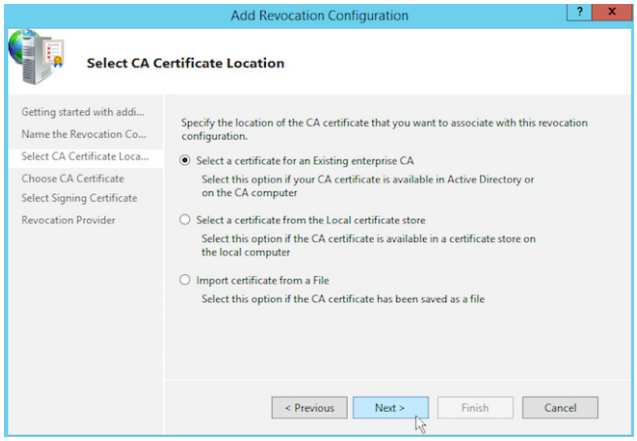
Right-click "Revocation Configuration," and select "Add Revocation Configuration." A configuration assistant appears.



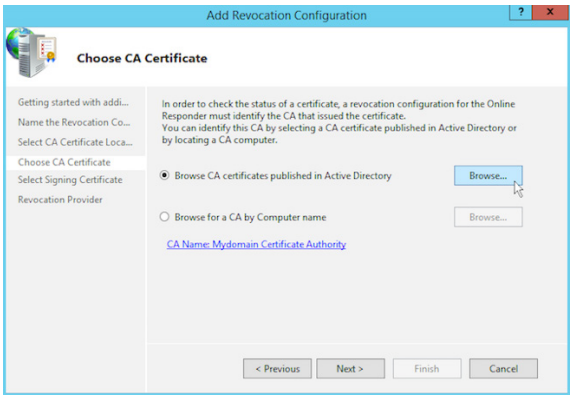
Click Next.



Enter a name and click Next.

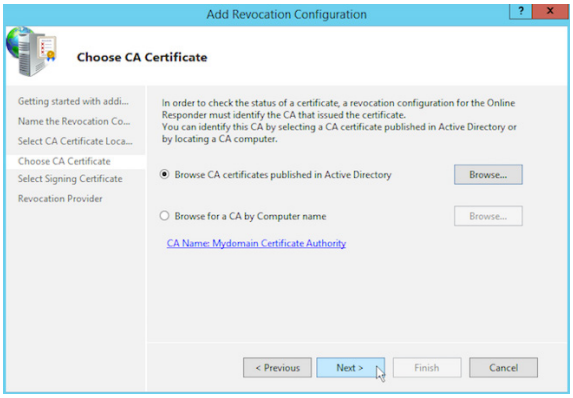


Select “Select a certificate for an Existing enterprise CA” and then click Next.

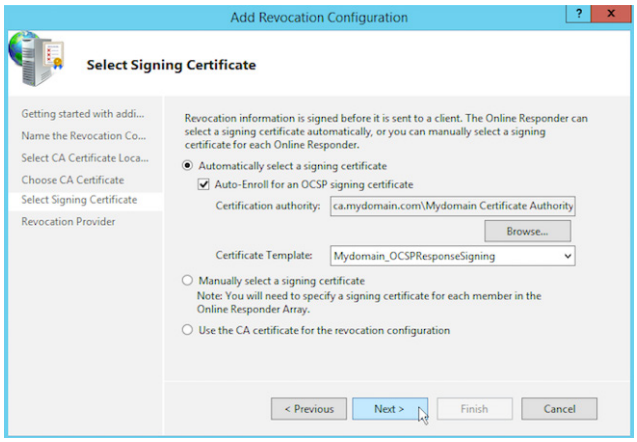


Click "Browse" to find your CA.

Select the CA Server you just created and click OK.

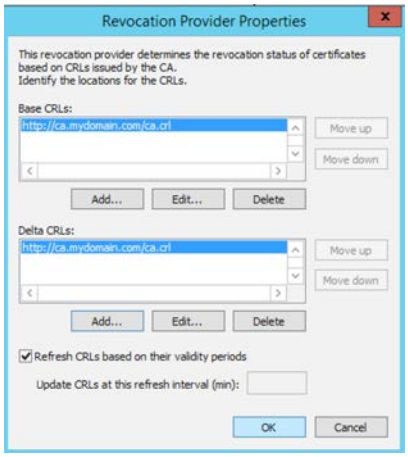


You should now see the name of the CA in blue text. Click Next.

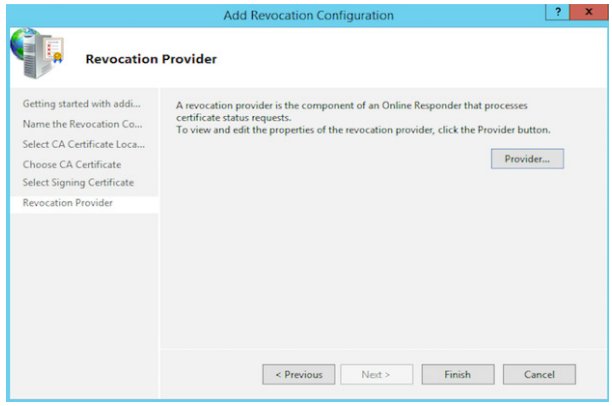


Normally the Certificate Authority and Certificate Template should come up automatically. If not, choose the "Browse" button to manually choose the Certificate Template, and then click Next.

In some cases, you get an error message. If this happens then manually add the provider list by clicking on the "Provider" check box.

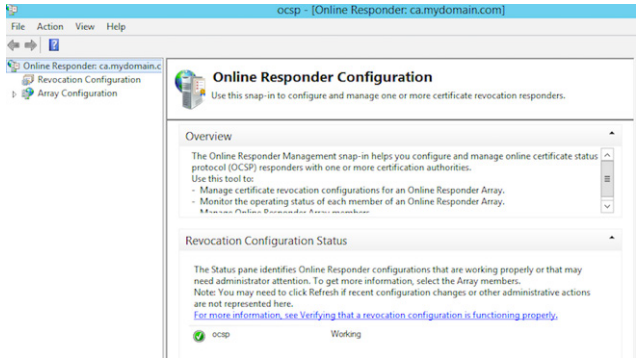


If you need to manually add the provider, you'll need to fill in this window as shown. If not, click Cancel and then click Finish.

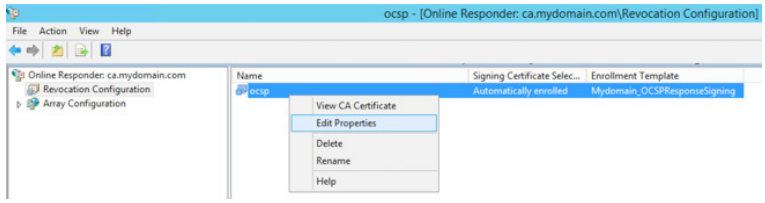


Click Finish.

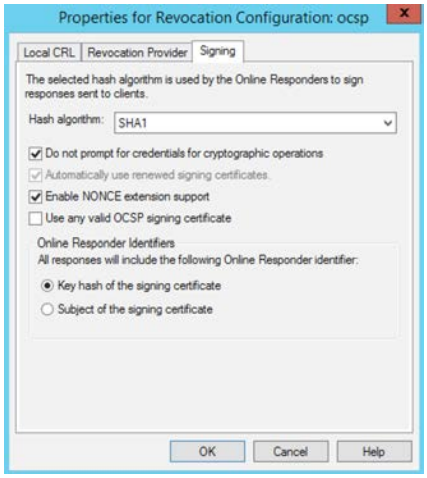
Now we need to change the revocation configuration properties.



Click on "Revocation Configuration" in the sidebar.

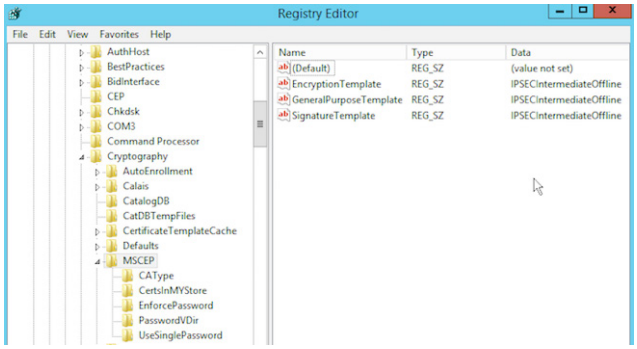


Click on the "Revocation Configuration," right-click "OCSP," and choose "Edit Properties."

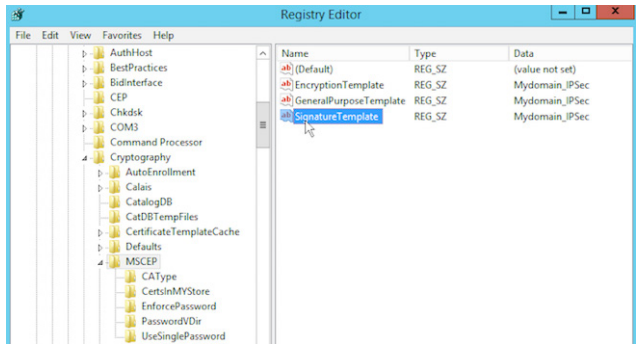


Click on the "Signing" tab and checkmark "Enable NONCE extension support," then click on OK. Now you are done with the OCSP configuration.

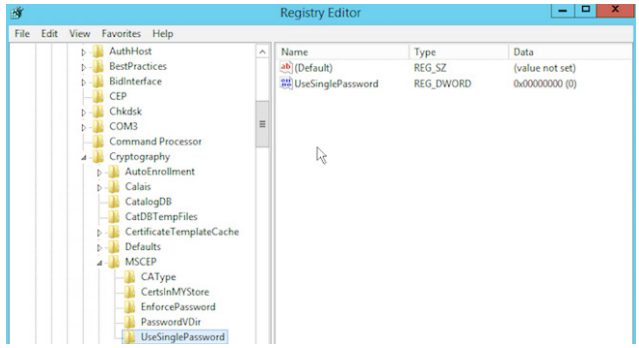
You should now enable SCEP with a permanent ticket or password. This is a security issue, but it's the only way to have a smooth, working solution that does not demand any hands-on attention when a certification is about to expire. Remember, it is not mandatory.



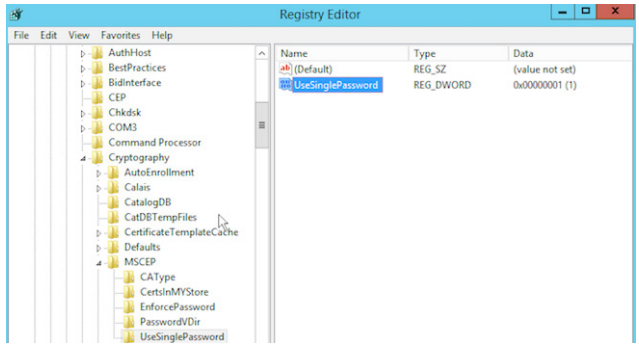
Now open the Registry Editor by typing "regedit" in the search function. Navigate to this path: *HKEY_LOCAL_MACHINE* > *SOFTWARE* > *Microsoft* > *Cryptography* > *MSCEP*.



You should now change the three data values from "IPSECIntermediateOffline" to the new name of the template you just created (we used "*Mydomain_IPsec*").

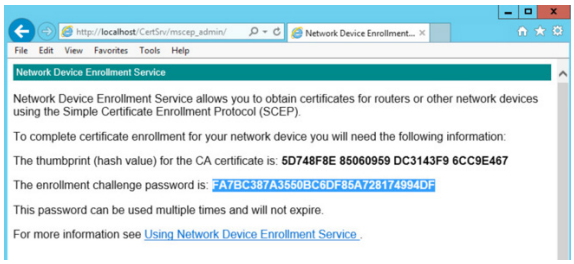


Navigate one step further to:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\MSCEP\UseSinglePassword.



And here change the data value from 0 to 1. Then you can close the Registry Editor.

Launch Internet Explorer or another browser.



Navigate to: http://localhost/CertSrv/mscep_admin. You'll need to have the credentials ready to manually log in, but that is not needed for your devices. Here you get the password (ticket) for device SCEP enrollments. The key for usage is shown here marked in blue.

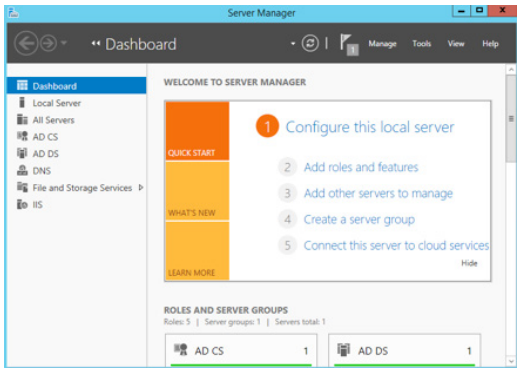
Okay, that's it! Now you're ready to enroll your devices through SCEP and verify the revocation status using OCSP.

Set up Microsoft Network Policy Server (NPS) on Windows Server 2012 R2 Std

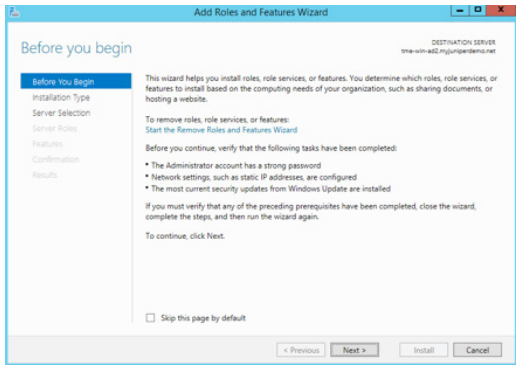
This is a step-by-step tutorial to help you set up Microsoft NPS and allow Remote Access Users to authenticate themselves using their Active Directory credentials or user certificate.

DISCLAIMER This section is only to show you how to set up Microsoft NPS on Windows Server 2012 R2 for a lab environment. It's not a best practice guide, and this system installation is not hardened. For a real, live scenario, please consult your Microsoft security expert.

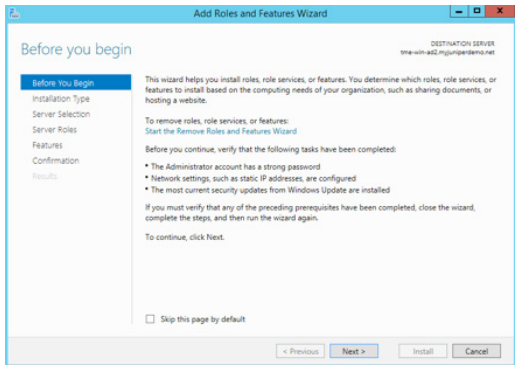
Go to the Dashboard.



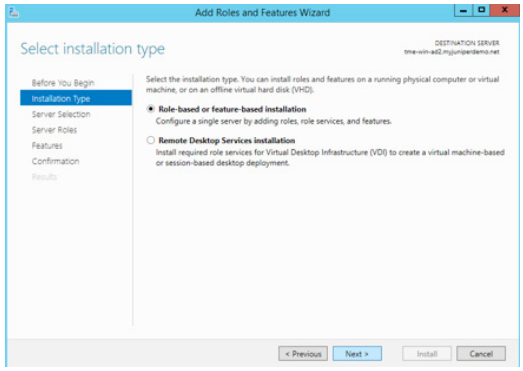
Click on number 2 "Add roles and features."



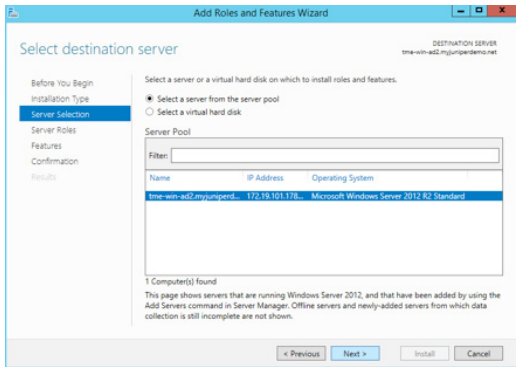
Click Next.



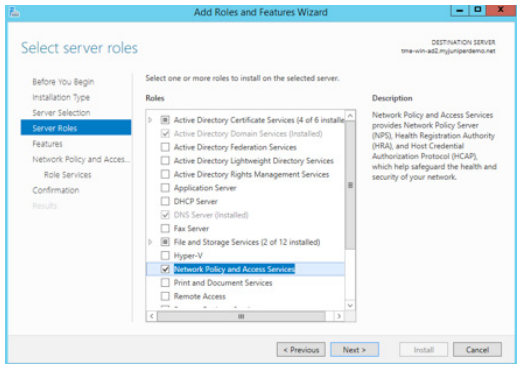
Click Next.



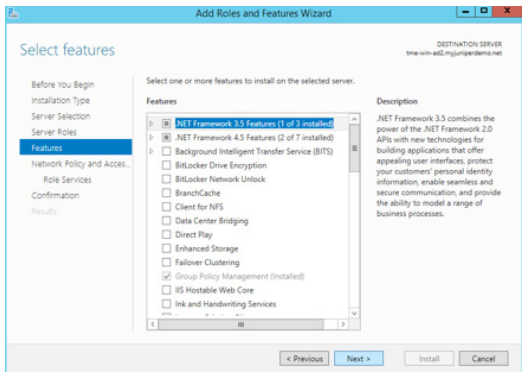
Click Next.



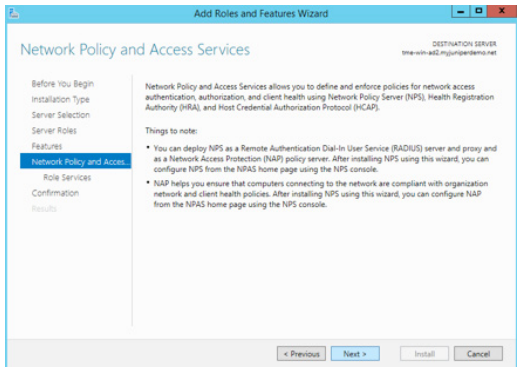
Click Next.



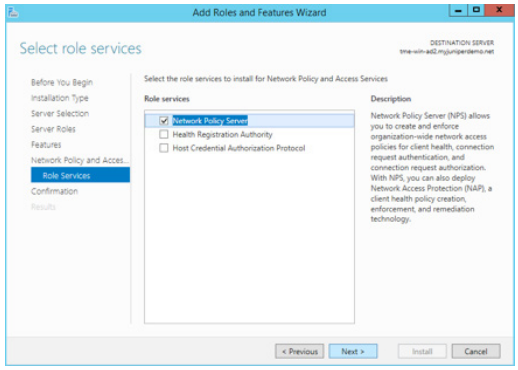
Check “Network Policy and Access Service” and this will pop-up a new window, Click “Add features” to close the pop-up window and then click Next.



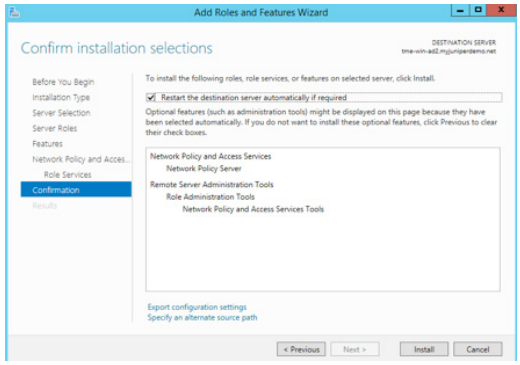
Click Next.



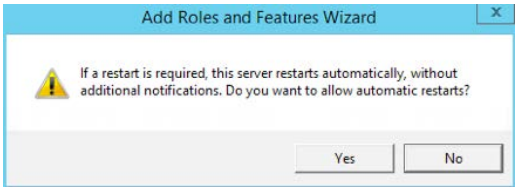
Click Next.



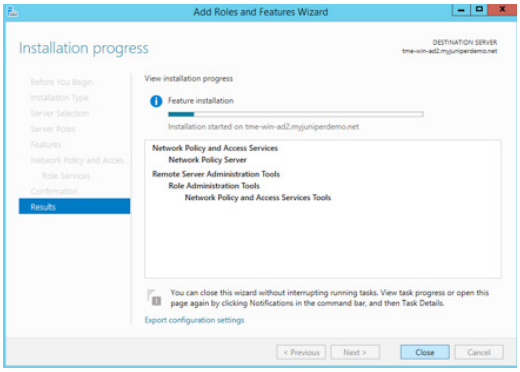
Click Next.



Check “Restart the destination server automatically if required.”



When you check "Restart the destination server automatically if required," it will pop up this window. Click "Yes" and then click "Next" on the previous screen.

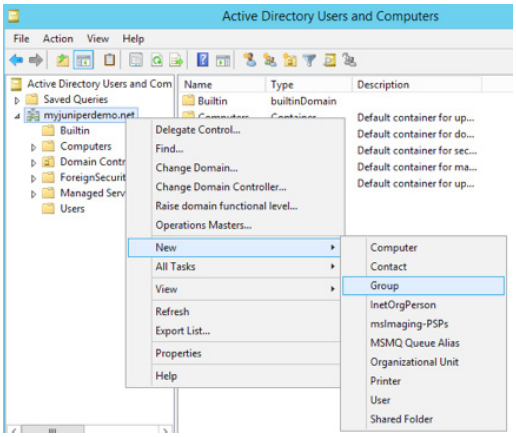


The system might restart, if so, click "Close" when it comes up and the blue progress bar shows that the installation is complete, or else wait and click "Close" when the installation is complete.

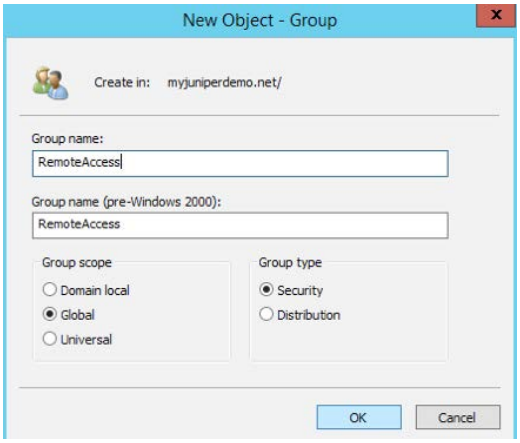
Generic RADIUS Configuration for Remote Access External Authentication

This section is a pre-requirement to enable external authentications. First, you should create a user group that is allowed to authenticate for remote access, and that will be used later in the set up process.

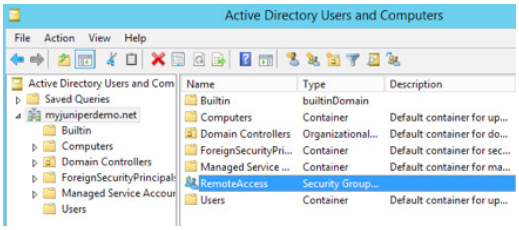
Open "Active Directory Users and Computers."



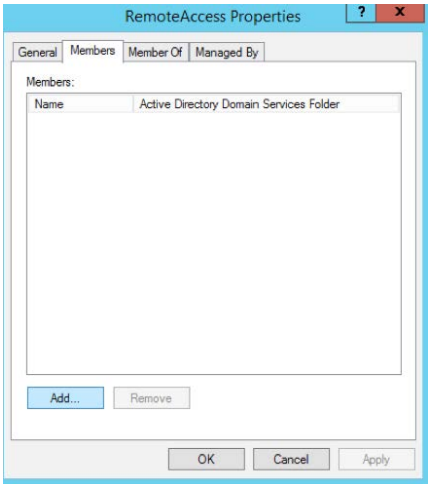
Right-click the appropriate domain and choose "New" then choose "Group" as shown here.



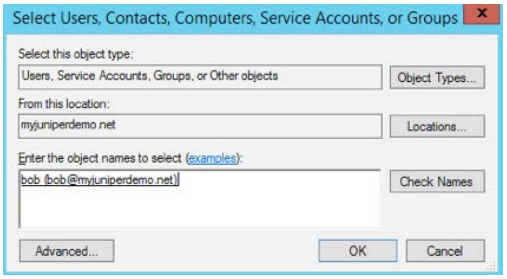
Give the group a name. The Group scope and Group type options are dependent on your environment. We checked Global and Security for our lab. When done click OK.



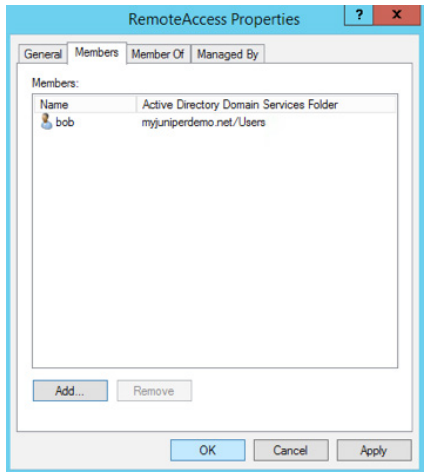
Now double-click on RemoteAccess to open it, so you can add "users" to this group.



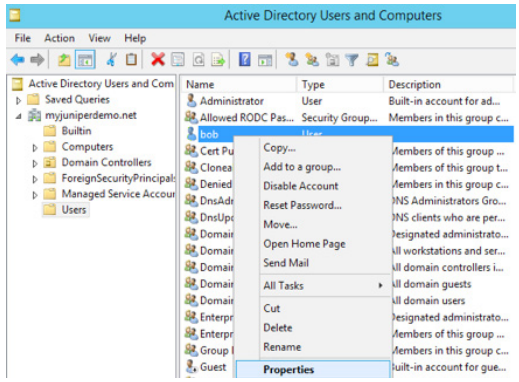
Click on the "Members" tab and click "Add."



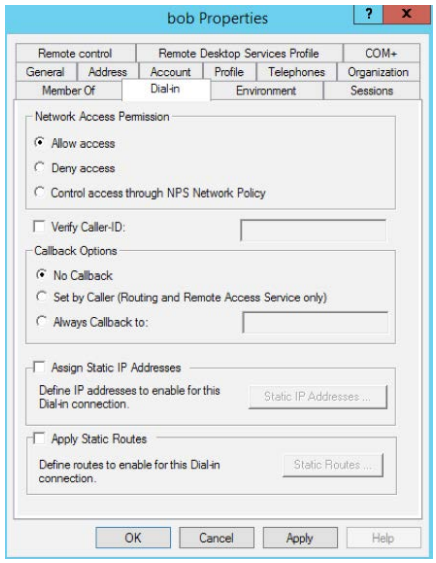
In the white box, type a username that should gain remote access, click "Check names." Our user is bob. Below is how it should look when the system finds bob. Add more users, or click OK.



Click OK.

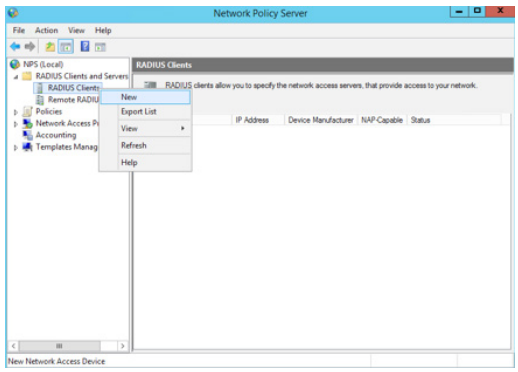


Before bob and other users get admission, you need to change the access rights per user. Navigate to the "Users" in the sidebar and right-click on each user and choose "Properties."

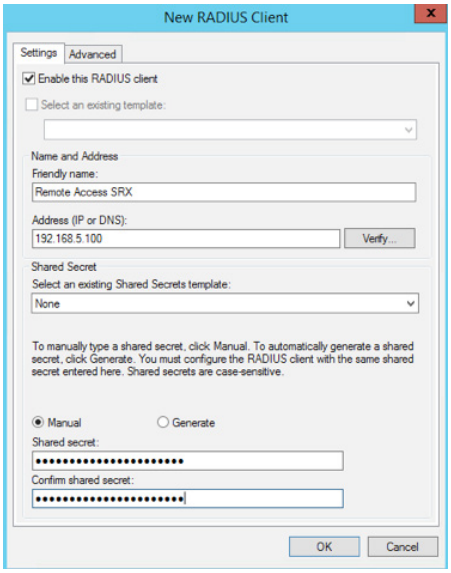


Now click on the "Dial-in" tab and check "Allow access" under the Network Access Permission section. Click OK.

Now open the Network Policy Server console. This is done by going to "Administrative tools" and choosing "Network Policy Server."



Then right-click on "RADIUS Client" and click "New."

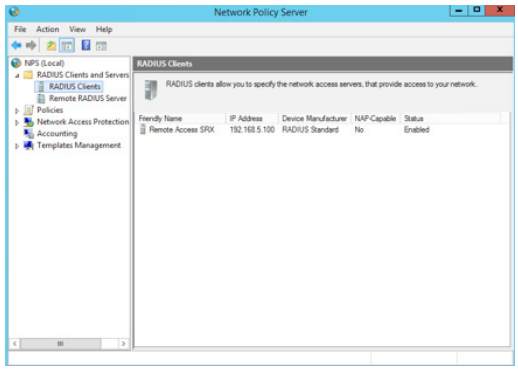


Give this RADIUS client a "Friendly name" that references your SRX. Here we use "Remote Access SRX."

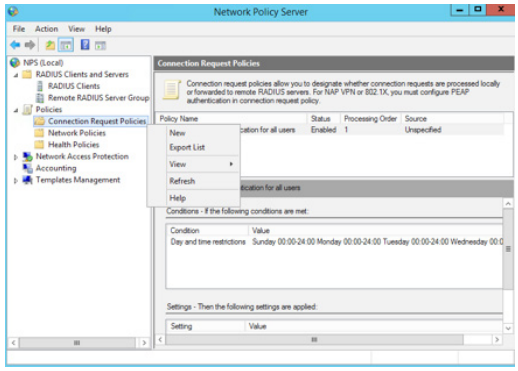
You also have to define the IP address that NPS will see from the SRX. As you only have one IP address on the internal interface on the SRX, let's enter this IP address as 192.168.5.100.

Finally, you must enter the same "Shared secret" that is used in the "Access Profile" on the SRX. It is recommended to have 22 characters with four classes where the shared secret starts and ends with a number or alpha character. Windows seems to have some challenges with certain passwords. This has been tested and worked for us: "1q2w3e4r5t6y7u8i9o0p!Q."

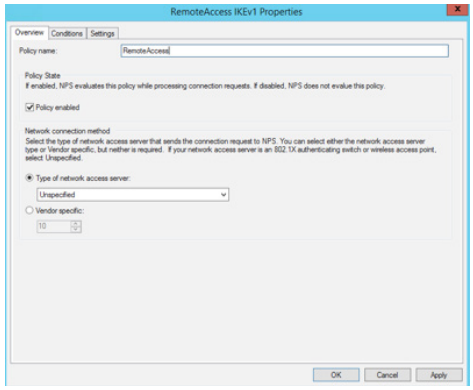
This how the RADIUS client config should look when you are done.



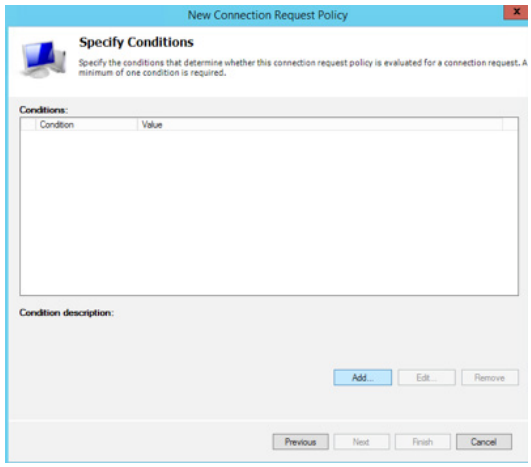
Now enable the users that should be allowed access.



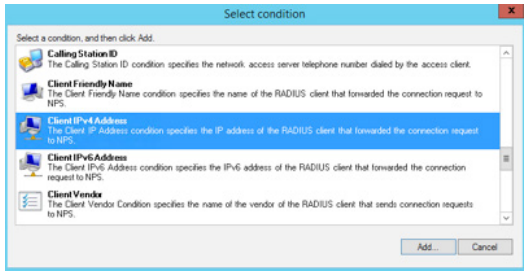
Right-click “Connection Request Policies” and click “New.”



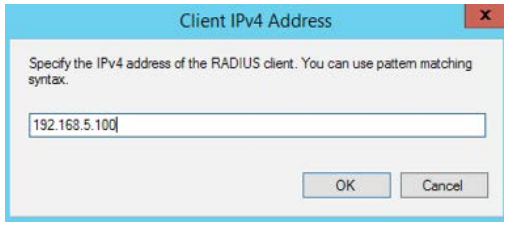
Enter a name in the Policy name section: we’re using “RemoteAccess” for this example. Click Apply.



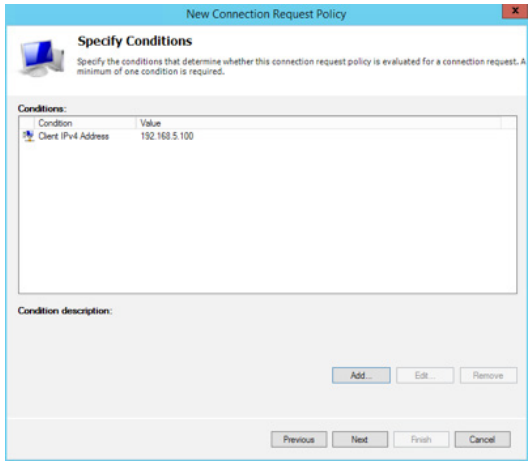
Click "Add."



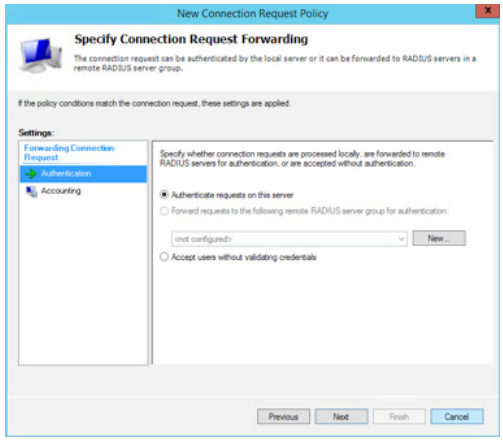
Navigate down until you find "Client IPv4 Address," select it and then click "Add."



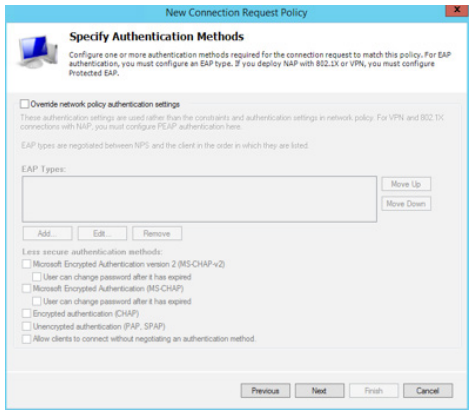
Below, enter the IP address of the SRX; this should be the same one you used when you defined the above RADIUS Client. We used 192.168.5.100. Click OK when finished.



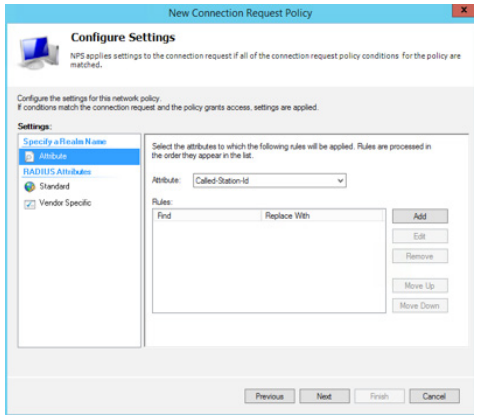
Click Next.



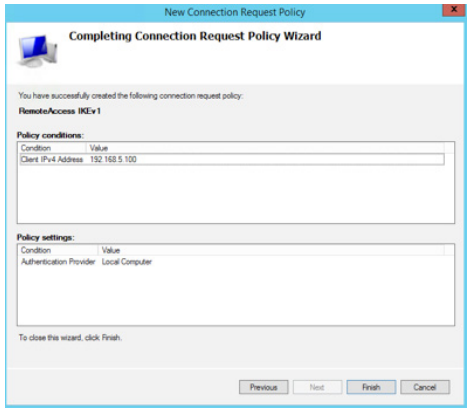
Click Next.



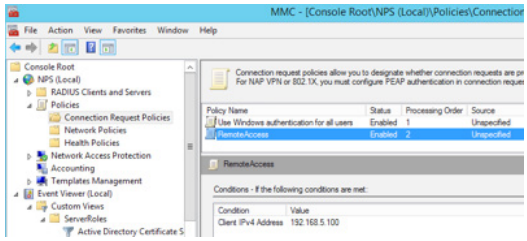
Click Next.



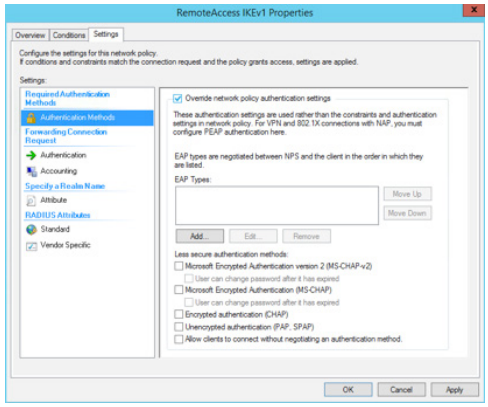
Click Next.



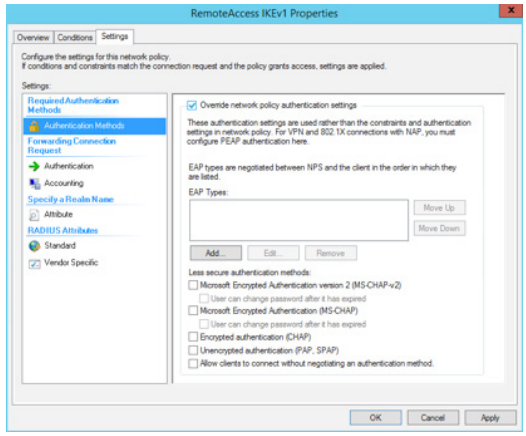
Click Finish.



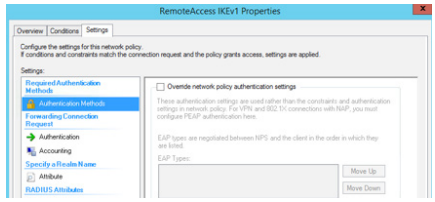
Highlight the newly created “Connection Request Policies” and double-click on the policy name to open up this configuration again.



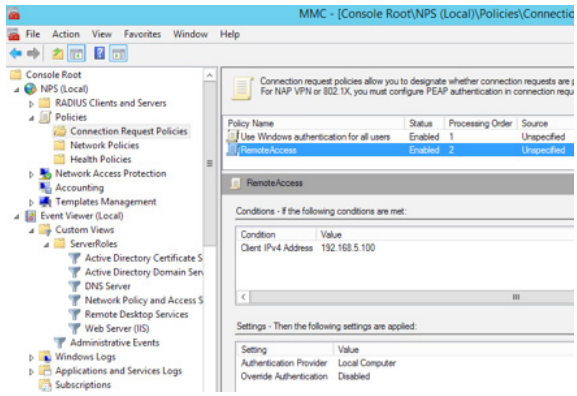
In this window, click on the “Settings” menu. Then check the “Override network policy authentication settings” option and click Apply. DO NOT click OK.



Uncheck the check box "Override network policy authentication settings." Now you can click OK.



Click OK.



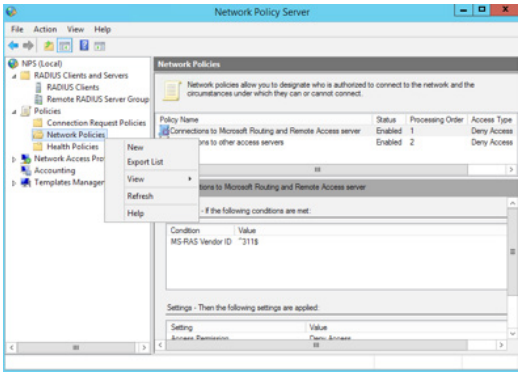
To verify, highlight the policy "RemoteAccess" and scroll to the bottom. You should see the following two settings: "Authentication Provider" and "Override Authentication."

Remote Access IKEv1 External Authentication (IKEv1)

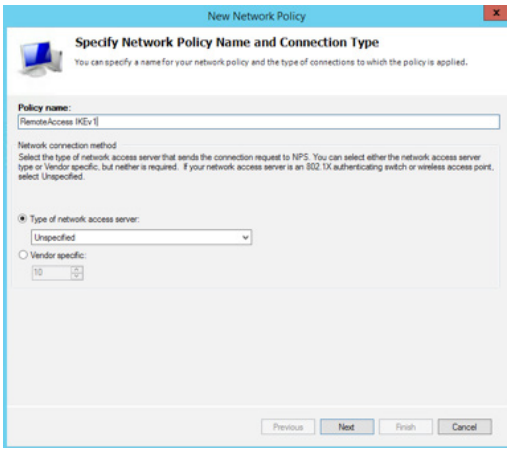
Now we're coming to the final steps for the RADIUS Authentication.

IMPORTANT Before attempting this tutorial step, you need to complete the previous sections of this chapter first: *NPS Installation*, and, *Generic RADIUS Configuration for Remote Access External Authentication*.

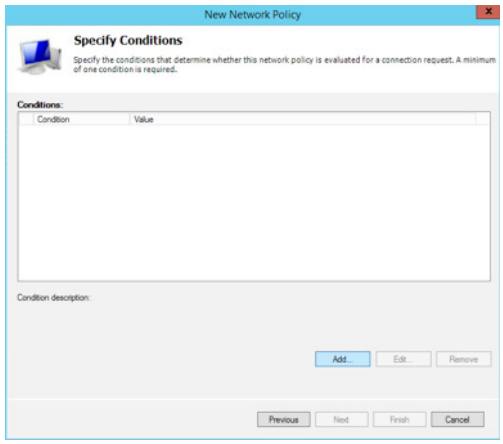
Navigate to the Network Policy Server.



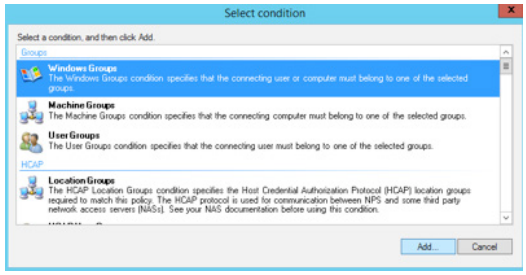
Right-click on "Network Policies" in the sidebar and choose "New."



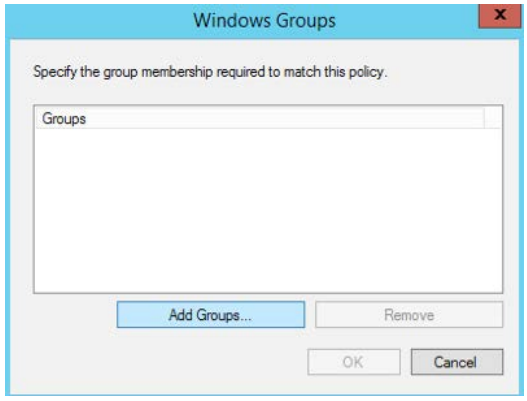
Give this policy a name, the same name as in "Connection Request Policies." We used "RemoteAccess IKEv1." When complete, click Next.



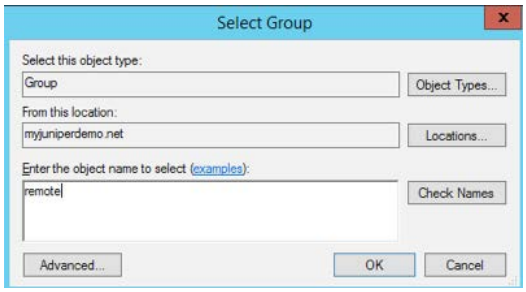
Click "Add."



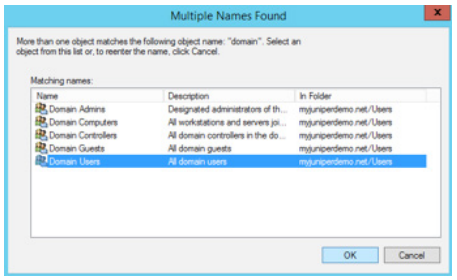
Highight "Windows Groups" and click "Add Groups."



Click "Add Groups."

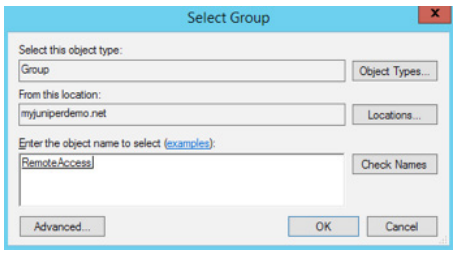


In our lab's case, we want to use the group "Remote Access." So we key in "remote" as the object name, and then click "Check Names" for the search.

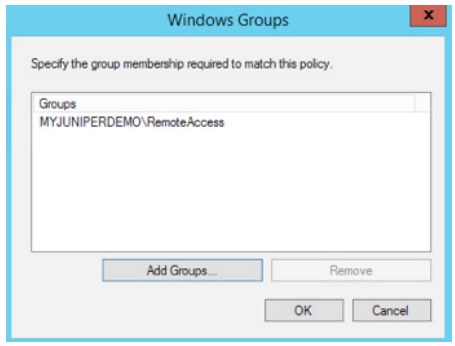


Depending on what you search for, you will get a list of objects. Click the appropriate one for your environment, and we will use "Remote Access" for our lab. Then highlight this group and click OK.

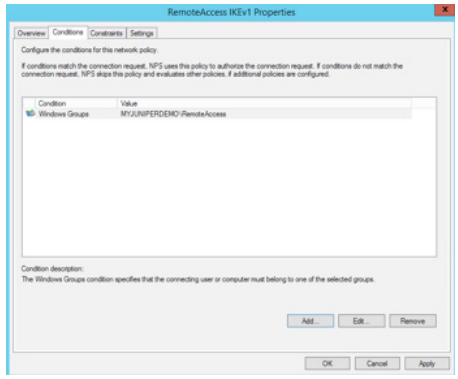
NOTE Depending on the group, it might resolve it directly; then go to the next step.



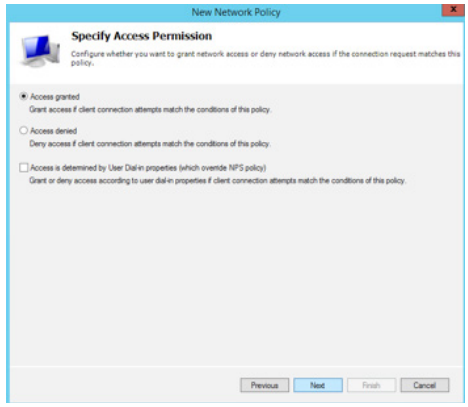
Click OK.



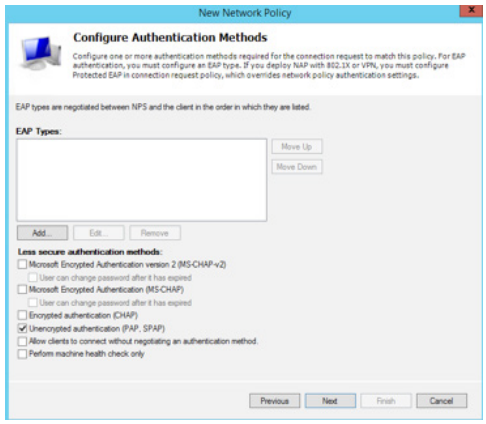
Click OK.



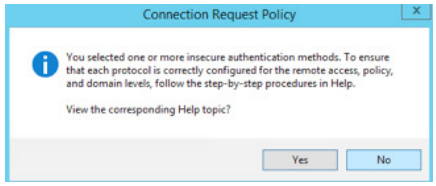
Click OK.



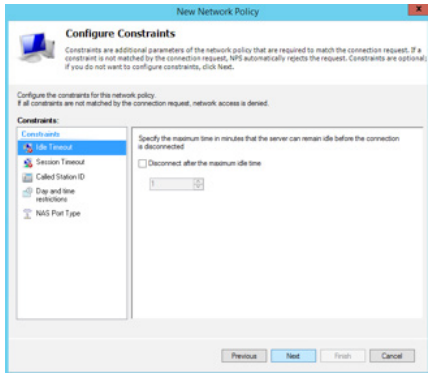
Select the "Access Granted" option and then click Next.



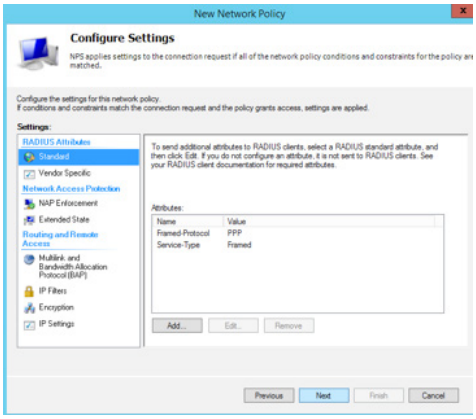
Check only the "Unencrypted authentication (PAP, SPAP)" option and then click Next.



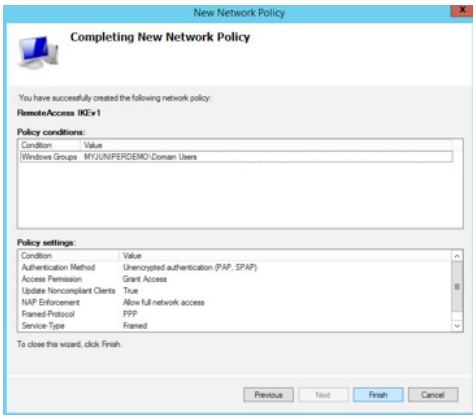
Click No.



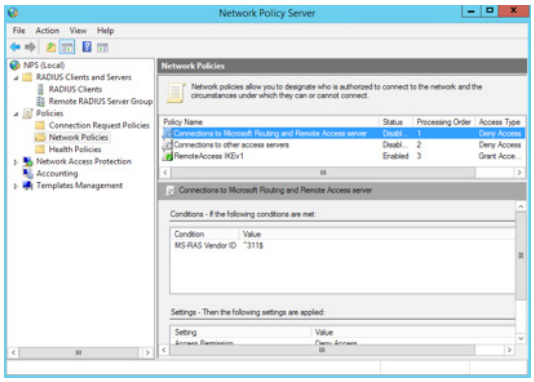
Click Next.



Click Next.



Click Finish.



And finally, right-click to disable "Connection to Microsoft Routing and Remote Access server" as well as "Connections to other access servers" network policies.

NOTE Keep in mind that this might be used in your system. So the reason why you might *not* be able to disable this is because it might impact other functions. In this case, please consult your Microsoft administrator before doing this.

Now your RADIUS server is configured to authenticate your users in the group "Remote Access" for IKEv1.

Let's proceed to Remote Access IKEv2.

Remote Access External Authentication (IKEv2) with EAP-TLS

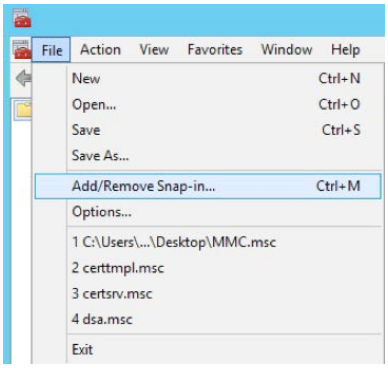
To use IKEv2 with EAP-TLS you need a certificate on the NPS (EAP-TLS) server as well as on the SRX Series, and one certificate for each user. In this section, you will learn how to request a certificate for the NPS (EAP-TLS) server on Microsoft. In Chapter 5 you will learn how to do this for both the SRX Series and the client.

These steps should be performed on the Microsoft server that hosts the NPS service.

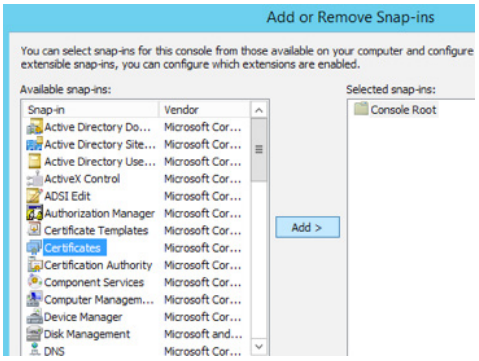
IMPORTANT Before attempting this tutorial, you need to complete the previous sections of this chapter: *NPS Installation*, and *Generic RADIUS Configuration for Remote Access External Authentication*.

Start by opening an MMC console or navigating through the Adminstrative tools section to come into the Certificate server section on the NPS server.

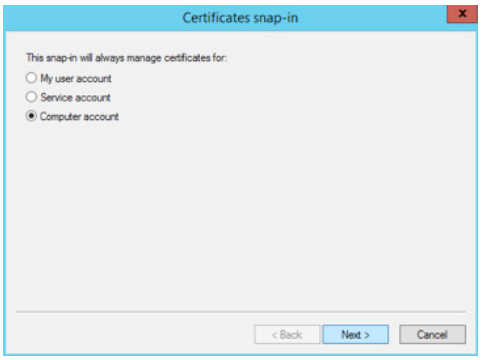
Open a clean MMC console using the magnifier and search for MMC.



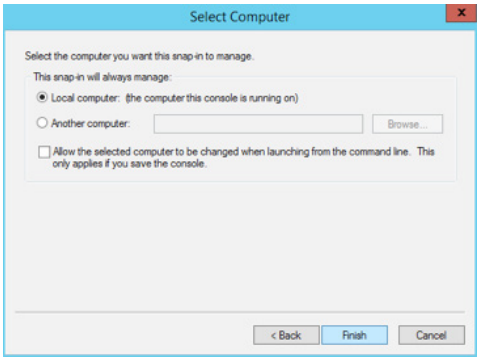
Click the menu "File" and choose "Add/Remove Snap-in."



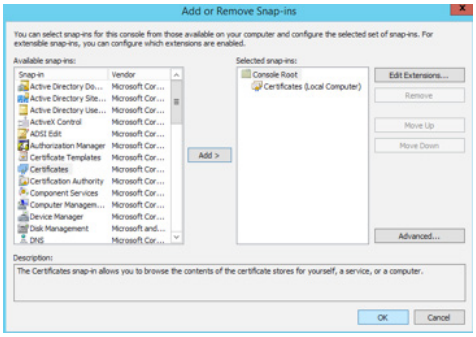
Highlight "Certificates" and click "Add."



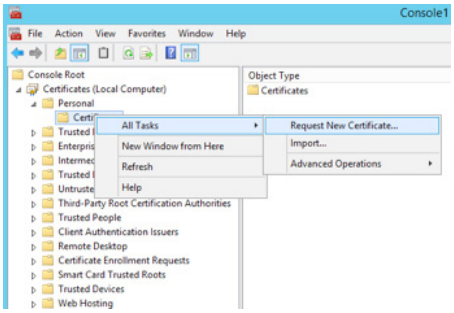
Check the "Computer account" option and click Next.



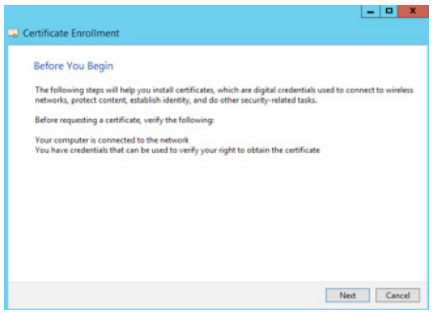
In this case, we'll use the same server, so just click Finish.



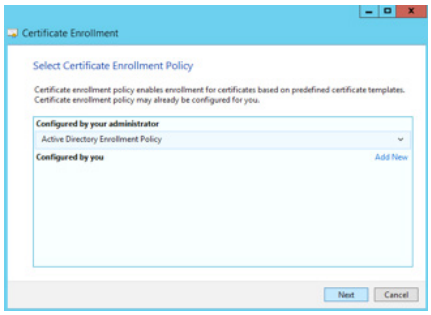
Click OK.



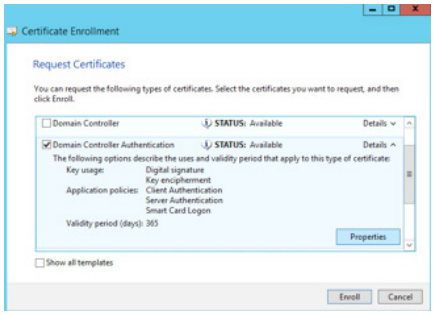
Follow the path *Console root* > *Certificates* > *Personal* > *Certificates* and right-click choosing "All Tasks" and click "Request New Certificate."



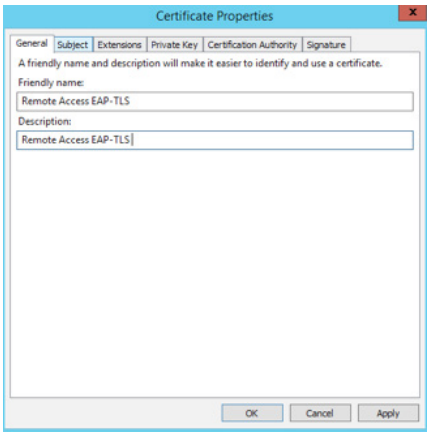
Click Next.



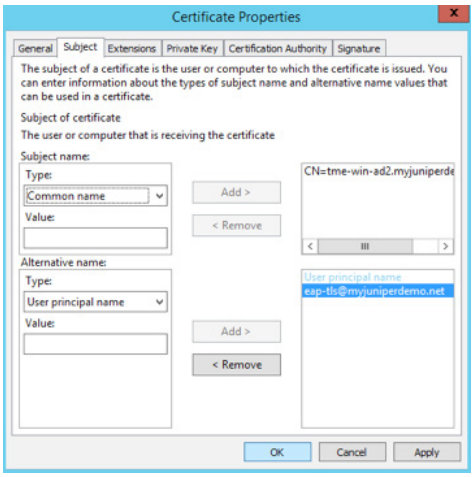
Highlight "Active Directory Enrollment Policy" and click Next.



Check the "Domain Controller Authentication" option and click "Properties."

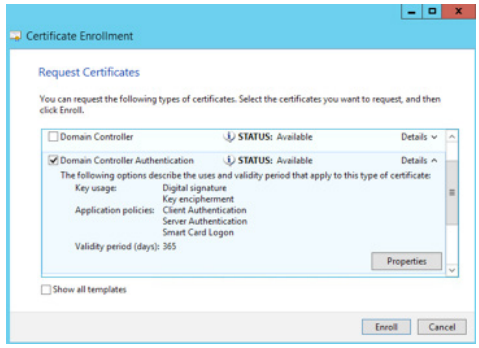


Give this certificate a "Friendly name" and "Description." Click on OK.

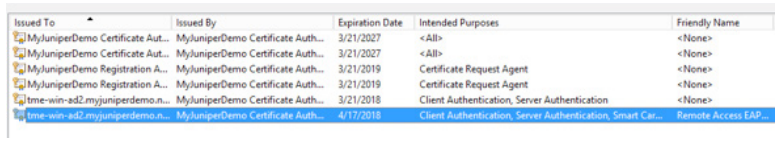


Click the "Subject" tab. For the "Subject name" add the FQDN of the host using the type "Common name." For the "Alternative name" add an email address that you want as a reference for the certificate using the type "User principal name," then click "OK."

In this lab's case: Subject name = 'Common name' "tme-win-ad2.myjuniperdemo.net"; and, Alternative name = 'User principal name' "eap-tls@myjuniperdemo.net."

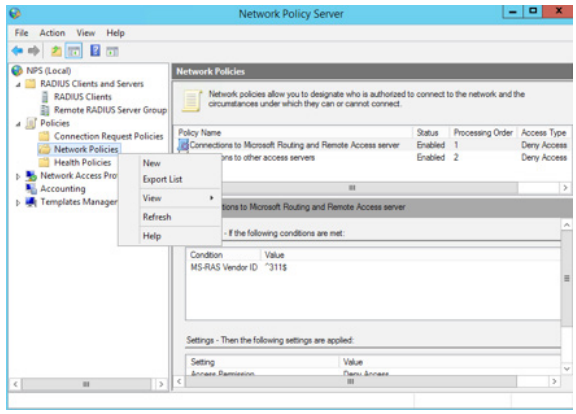


Click Enroll.

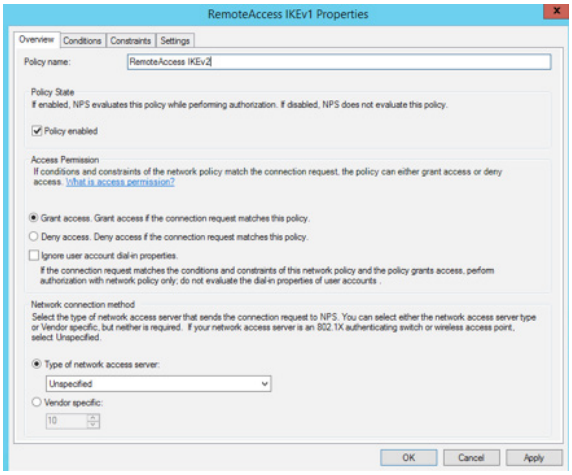


You should now find the following certificate in your certificate store.

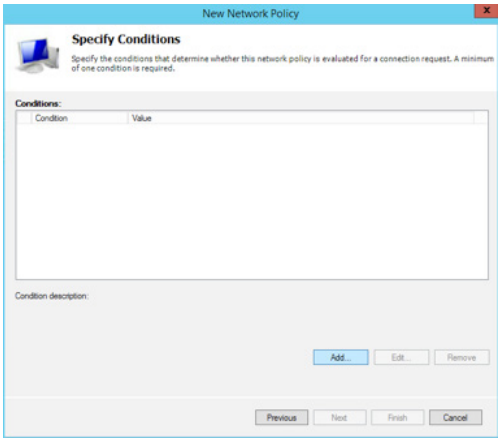
Now we're coming to the final steps for the RADIUS Authentication, but first back to the Network Policy Server.



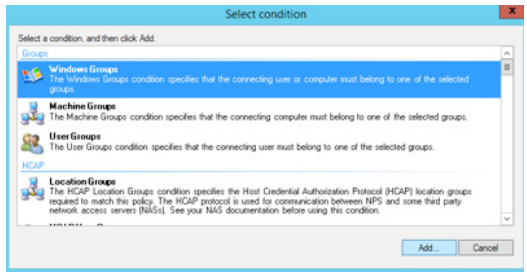
Right-click "Network Policies" in the sidebar and choose "New."



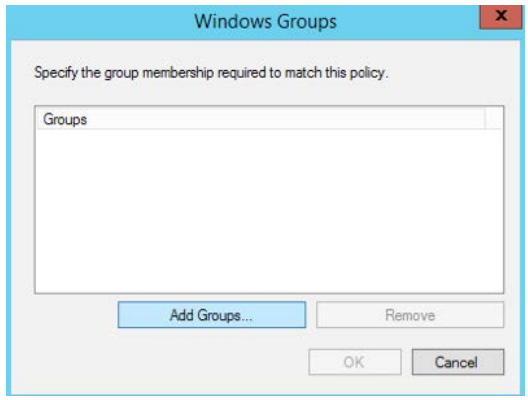
Give this Policy Name the same name as the "Connection Request Policies." Our lab uses "RemoteAccess IKEv2." When finished click OK.



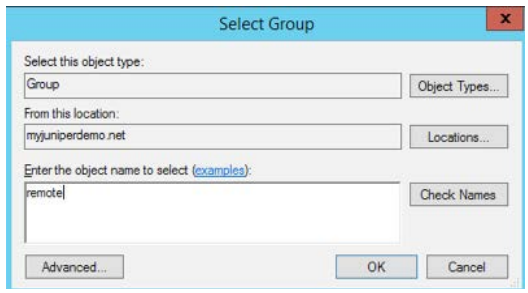
Click "Add."



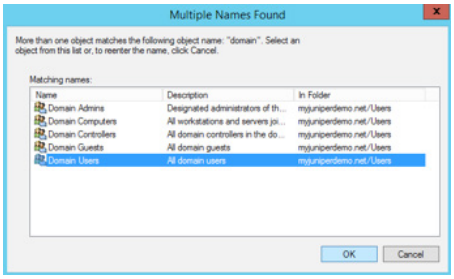
Highlight "Windows User Groups" and click "Add."



Click "Add Groups."

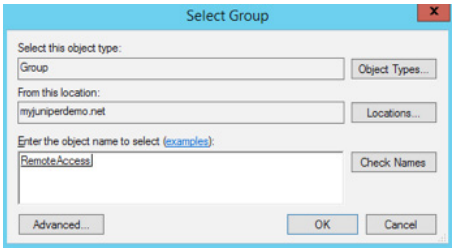


In our case, we would like to use the group "Remote Access" so we key in "remote" for the search and click "Check Names."

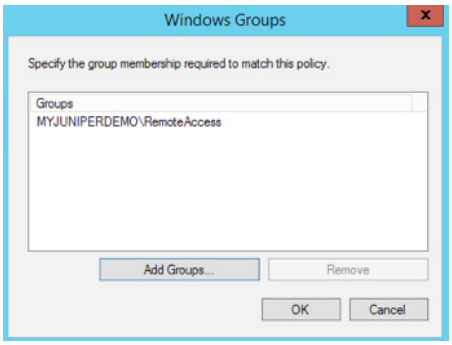


Depending on what you search for, you will get a list of objects; click the appropriate ones for your environment. Then highlight this group and click OK.

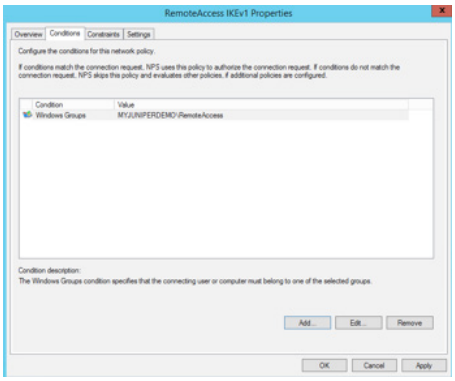
Depending on the group, it might resolve directly; then go to the next step.



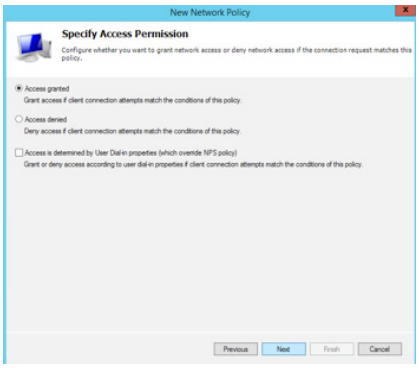
Click OK.



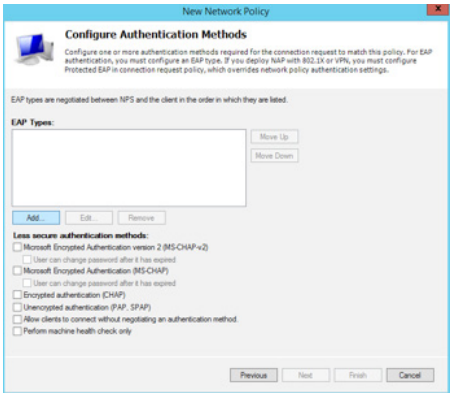
Click OK.



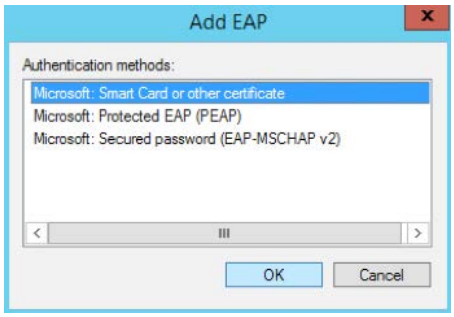
Click OK.



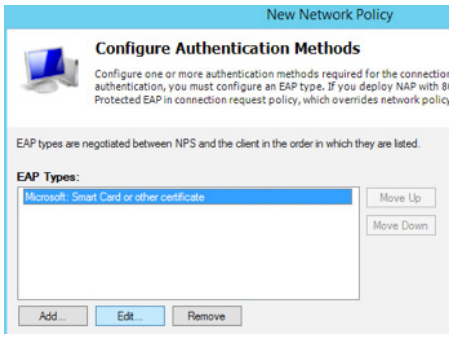
Check the "Access Granted" option and then click Next.



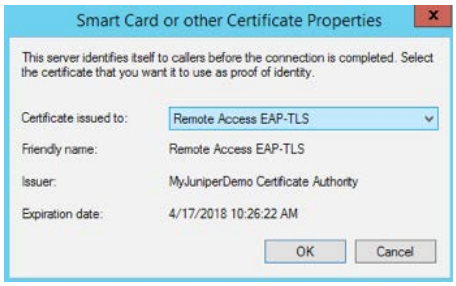
Uncheck all options under "Less secure authentication methods." Click "Add."



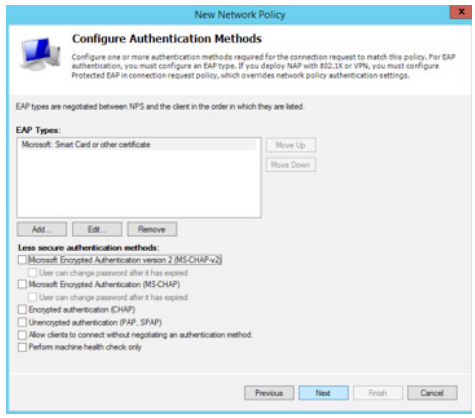
Highlight "Microsoft: Smart Card or other certificate" then click OK.



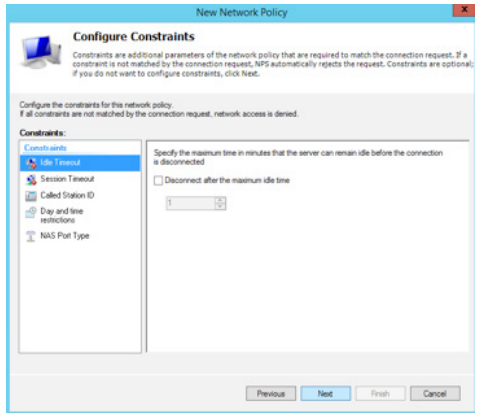
Now highlight the newly-added EAP Type and click Edit.



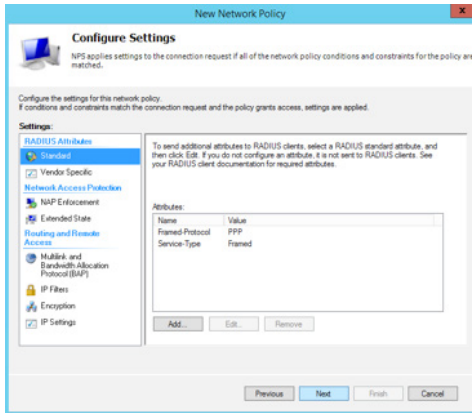
Verify that it uses the previously created certificate and click OK, or else change to the correct certificate, before clicking OK.



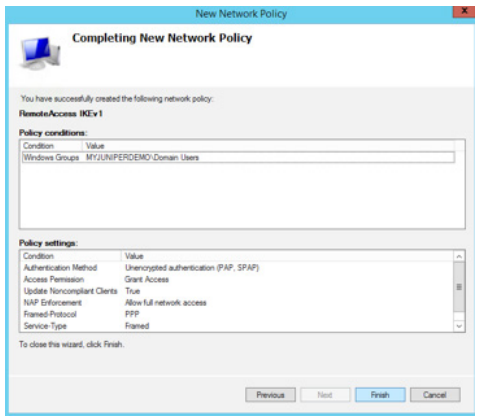
Click Next.



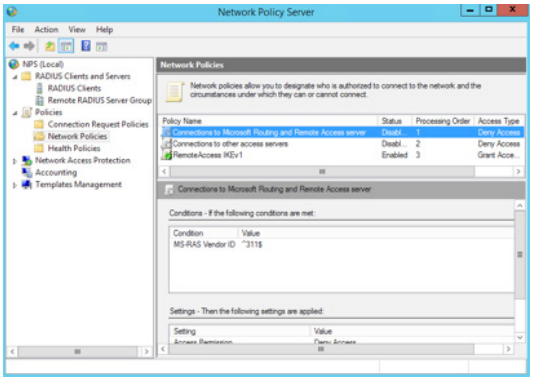
Click Next.



Click Next.



Click Finish.



And finally, right-click to disable "Connection to Microsoft Routing and Remote Access server" as well as "Connections to other access servers" network policies.

NOTE Keep in mind that this might be used in your system. So the reason why you might *not* be able to disable this is because it might impact other functions. In this case, please consult your Microsoft administrator before doing this.

Now your RADIUS server is configured to authenticate your users in the group "Remote Access" for IKEv2.

Chapter 3

Pick a Site-to-Site Configuration

IPSec chapters

There is no need for a lengthly introduction here. Find your site-to-site configuration needs in the many examples that follow.

One note to the reader: whenever the Junos requirements say "<Junos version> *is used*" this means that it might work in releases below that version, if the state is "<Junos version> *and above*" this mean it is supported beginning with that release.

Site-to-Site Using Static Peers

Using static peers means you have two locations that should establish an IPsec tunnel using Certificate-based authentication. See Figure 3.1.

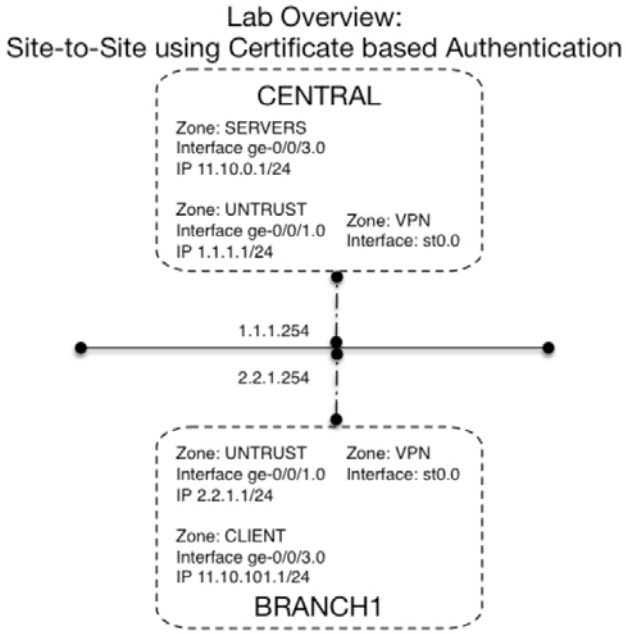


Figure 3.1 Site-to-Site Using Static Peers

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming ssh access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet
root@host# set interface st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Configure a route for the remote side:

```
root@host# set routing-options static route 11.10.101.0/24 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone plus allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Now it's time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration to verify our IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority
root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE To reduce log size in a production environment, you can remove the `session-init` statement for logging.

First, build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

```
root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Somehow you need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:    X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to the URL: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:    X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status:
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a certificate and using a strong authentication and encryption algorithm with a rekey time of 28800 seconds:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells you at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is our lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

Okay! It's time to commit and wait for the other spoke to be configured so you can verify the topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

root@host# set system root-authentication plain-text-password

```
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet
root@host# set interface st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure a route to the remote side:

```
root@host# set routing-options static route 11.10.0.0/24 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top
```

Keep in mind that the `challenge-password` below is unique to your CA server:

```
root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you somehow need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

You should verify that you trust the downloaded certificate:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=Branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status:
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

This example uses Site-to-Site as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Let's also add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the config:

```
root@BRANCH1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the troubleshooting section in this chapter. Each local site should have a client from which you send traffic if you want to verify that traffic is floating through the system, otherwise you need to configure local policies to allow the Junos host to send traffic between certain zones.

On Central

The following show command confirms that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 2
 ID        Algorithm    SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  1b294297 1623/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  d5205ec5 1623/ unlim  - root 500  2.2.1.1
```

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  7204390a 3303/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  53a506ec 3303/ unlim  - root 500  1.1.1.1
```

Site-to-Site with Source NAT Inside Tunnel

In this case, you have a remote location that needs to hide its internal IPs using a public IP of 5.5.5.20. This could be, for example, when you don't want to expose the branch side's internal addressing or the central site does not allow a remote location coming in with a black IP (RFC1918) address (this can of course be done for the opposite direction). Keep your routing and firewall policies in mind when executing changes. When testing this case, do it from a host on the trusted side of the network, or else you will need more policies and changes to the NAT configuration that maps to the Junos-host zone.

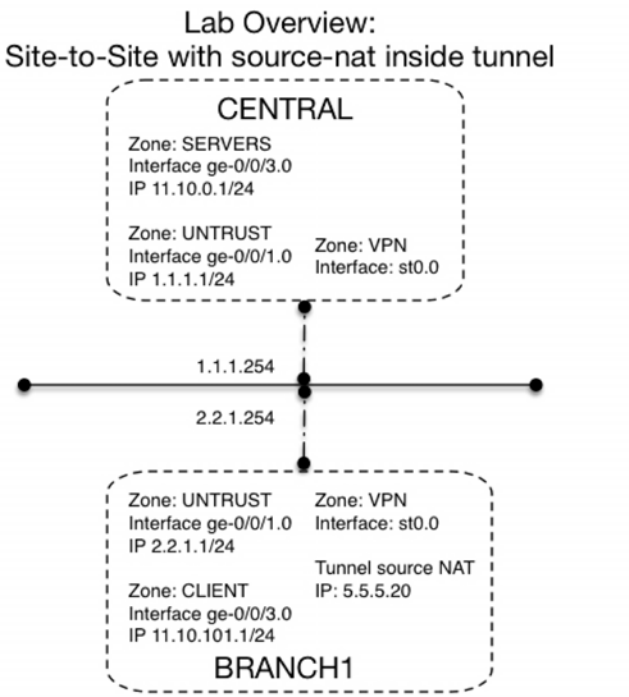


Figure 3.2 Site-to-Site with Source NAT Inside Tunnel

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate Authority: SCEP (Simple Certificate Enrollment Protocol) and

OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware versions that support the above-referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work properly:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configuration, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

As the remote side will NAT traffic, and coming from IP 5.5.5.20, you need a route back to that network:

```
root@host# set routing-options static route 5.5.5.20/32 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

It's now time to configure and request the certificates that you need to establish the tunnel. First you need to define how to find the Certificate Authority:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Specify the Certificate Authority SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration to verify our IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority

root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address 5.5.5.20/32 5.5.5.20/32
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic should be allowed in different direction.

NOTE In a production environment, you can remove the `session-init` statement for logging to reduce log size.

First is to build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top

root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address 5.5.5.20/32
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Somehow you need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

By running the below command, you can see the current numbers of valid OCSP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to URL http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status:
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple

tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells me at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish an Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured before you can verify our topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure a route pointing to the central site:

```
root@host# set routing-options static route 11.10.0.0/24 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Here we specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@host# run ping 2.2.1.254
root@host# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

First we define our address book objects that should be used in our security policys:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Let's define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you somehow need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service.

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=Branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status:
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Source NAT

As the central side does not want to handle multiple clients that might have the same subnets, we need to hide our internal networks behind a different IP than the original:

```
root@BRANCH1# edit security nat source
root@BRANCH1# set pool IPSEC address 5.5.5.20/32
root@BRANCH1# set rule-set IPSEC from zone CLIENTS
root@BRANCH1# set rule-set IPSEC to zone VPN
root@BRANCH1# set rule-set IPSEC rule 1 match source-address 0.0.0.0/0
root@BRANCH1# set rule-set IPSEC rule 1 then source-nat pool IPSEC
root@BRANCH1# top
```

And add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the config:

```
root@BRANCH1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the Troubleshooting section of the book. Each local site should have a client from which you send traffic if you want to verify that traffic floating through the system, otherwise you need to configure local policies to allow junos-host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  c7ad4208 3242/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  9e9e0b6  3242/ unlim  - root 500  2.2.1.1
```

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  9e9e0b6  3297/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  c7ad4208 3297/ unlim  - root 500  1.1.1.1
```

As you NAT the traffic from this side, you can verify that you actually hit the NAT rule base:

```
root@BRANCH1# run show security nat source rule all
Total rules: 1
Total referenced IPv4/IPv6 ip-prefixes: 1/0
source NAT rule: 1    Rule-set: IPSEC
 Rule-Id:      2
 Rule position:  1
```

```
From zone:  CLIENTS
To zone:      VPN
Match
Source addresses:    0.0.0.0 - 255.255.255.255
Action:       IPSEC
Persistent NAT type: N/A
Persistent NAT mapping type:  address-port-mapping
Inactivity timeout:    0
Max session number:  0
Translation hits:    1
Successful sessions:   1
Failed sessions:   0
Number of sessions:    1
```

Site-to-Site with Source NAT Using Public IP

In this case with Source NAT using Public IP, there is a remote location that needs to hide its internal IPs using their public IP address 2.2.1.1. Because they only have one public IP to use and their ISP doesn't offer anymore IPs, this could be an example for when the central side does not allow remote location to log in with a black IP (RFC1918) address. This can, of course, be done for the opposite direction. Keep in mind routing and firewall policies when doing these changes. When testing this, do it from a host on the Trusted side of the network, or else you need more policies and changes to the NAT configuration that maps to the Junos-host zone.

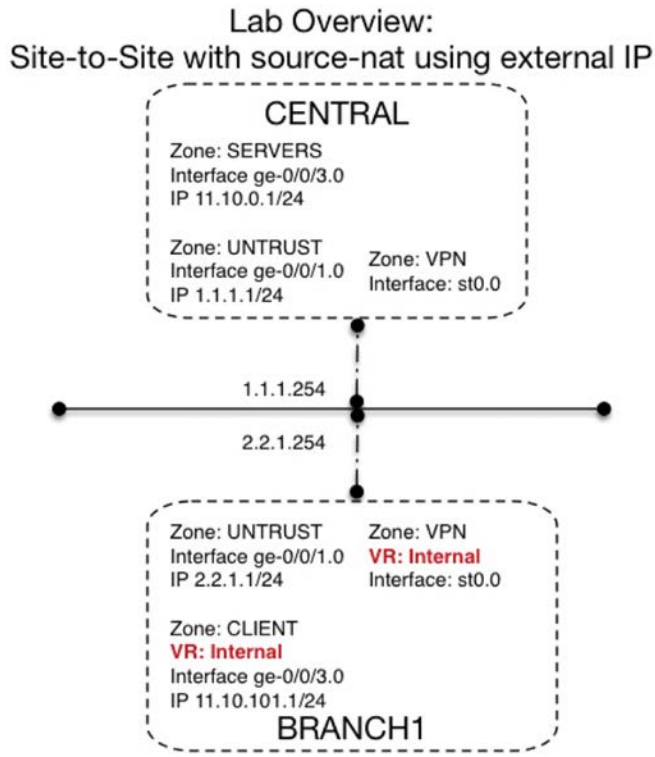


Figure 3.3 *Site-to-Site with Source NAT Using Public IP*

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate Authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this cookbook. Keep in mind that this cookbook has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configuration, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

To be able to reply from source NAT with the external IP of the remote IKE peer, you need to place the internal and ST interface into a Virtual Router (VR). You also have to build RIB groups so traffic can reach this routing instance:

```
root@host# edit routing-instances INTERNAL
root@host# set instance-type virtual-router
root@host# set interface ge-0/0/3.0
root@host# set interface st0.0
root@host# set routing-options static route 2.2.1.1/32 next-hop st0.0
root@host# set routing-options interface-routes rib-group inet INTERNAL_TO_INET
root@host# set routing-options static rib-group INTERNAL_TO_INET
root@host# top

root@host# edit routing-options rib-groups
root@host# set INET_TO_INTERNAL import-rib inet.0
root@host# set INET_TO_INTERNAL import-rib INTERNAL.inet.0
root@host# set INET_TO_INTERNAL import-policy INET_TO_INTERNAL
root@host# set INTERNAL_TO_INET import-rib INTERNAL.inet.0
root@host# set INTERNAL_TO_INET import-rib inet.0
root@host# set INTERNAL_TO_INET import-policy INTERNAL_TO_INET
root@host# set INTERNAL_TO_INET import-rib INTERNAL.inet.0
root@host# set INTERNAL_TO_INET import-rib inet.0
root@host# set INTERNAL_TO_INET import-policy INTERNAL_TO_INET
root@host# up
root@host# set interface-routes rib-group inet INET_TO_INTERNAL
root@host# set static rib-group INET_TO_INTERNAL
root@host# top

root@host# set policy-options prefix-list INTERNAL_TO_INET_REJECT 2.2.1.1/32
root@host# edit policy-options policy-statement INET_TO_INTERNAL
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top

root@host# edit policy-options policy-statement INTERNAL_TO_INET
root@host# set term REJECT from protocol static
root@host# set term REJECT from prefix-list INTERNAL_TO_INET_REJECT
root@host# set term REJECT then reject
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Here specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration to verify our IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority

root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address IPSEC-2.2.1.1 2.2.1.1/32
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic that should be allowed in different directions.

NOTE In a production environment, you can remove the session-init statement for logging to reduce log size.

First is to build the policy that allows incoming traffic to the CA server from our remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
```

```
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top

root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address IPSEC-2.2.1.1
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Somehow you need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now let's verify that we trust the downloaded certificate. When running the show command, you can see the current numbers of valid OCSP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give it a unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Where:

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to the URL http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells us at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
```

```
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured so you can verify the topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

To be able to reply from source NAT with the external IP of the remote IKE peer, you need to place the internal and ST interface into a `virtual-router`. You also have to build RIB groups so traffic can reach this routing instance:

```
root@host# edit routing-instances INTERNAL
root@host# set instance-type virtual-router
root@host# set interface ge-0/0/3.0
root@host# set interface st0.0
root@host# set routing-options static route 11.10.0.0/24 next-hop st0.0
root@host# set routing-options interface-routes rib-group inet INTERNAL_TO_INET
root@host# set routing-options static rib-group INTERNAL_TO_INET
root@host# top

root@host# edit routing-options rib-groups
root@host# set INET_TO_INTERNAL import-rib inet.0
root@host# set INET_TO_INTERNAL import-rib INTERNAL.inet.0
root@host# set INET_TO_INTERNAL import-policy INET_TO_INTERNAL
root@host# set INTERNAL_TO_INET import-rib INTERNAL.inet.0
root@host# set INTERNAL_TO_INET import-rib inet.0
root@host# set INTERNAL_TO_INET import-policy INTERNAL_TO_INET
root@host# up
root@host# set interface-routes rib-group inet INET_TO_INTERNAL
root@host# set static rib-group INET_TO_INTERNAL
root@host# top
```

```
root@host# edit policy-options policy-statement INET_TO_INTERNAL
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top

root@host# set policy-options prefix-list INTERNAL_TO_INET_REJECT

root@host# edit policy-options policy-statement INTERNAL_TO_INET
root@host# set term REJECT from protocol static
root@host# set term REJECT from prefix-list INTERNAL_TO_INET_REJECT
root@host# set term REJECT then reject
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing any incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to commit this config to verify our IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach the gateway and also that the remote gateway is using ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you somehow need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
```

```
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the certificate.

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server
CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status
```

By running is command, you can see that if you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=Branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
```

```
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter. This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
```

```
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Source NAT

As the central side does not want to handle multiple clients that might have the same subnets, you need to hide your internal networks behind the external IP as you only have one IP assigned to you:

```
root@BRANCH1# edit security nat source
root@BRANCH1# set pool IPSEC address 2.2.1.1/32
root@BRANCH1# set rule-set IPSEC from zone CLIENTS
root@BRANCH1# set rule-set IPSEC to zone VPN
root@BRANCH1# set rule-set IPSEC rule 1 match source-address 0.0.0.0/0
root@BRANCH1# set rule-set IPSEC rule 1 then source-nat pool IPSEC
root@BRANCH1# top
```

Add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the config:

```
root@BRANCH1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the troubleshooting section under site-to-site. Each local site should have a client from which you send traffic if you want to verify that traffic floating through the system, otherwise you need to configure local policies to allow junos-host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  c7ad4208 3242/ unlim  - root 500 2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  9e9e0b6  3242/ unlim  - root 500 2.2.1.1
```

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  9e9e0b6  3297/ unlim  - root 500 1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  c7ad4208 3297/ unlim  - root 500 1.1.1.1
```

As you should NAT your traffic from this side, you can verify that you actually hit the NAT rule base:

```
root@BRANCH1# run show security nat source rule all
Total rules: 1
Total referenced IPv4/IPv6 ip-prefixes: 1/0
source NAT rule: 1    Rule-set: IPSEC
 Rule-Id:      2
 Rule position:  1
 From zone:  CLIENTS
 To zone:      VPN
 Match
 Source addresses:    0.0.0.0 - 255.255.255.255
 Action:       IPSEC
 Persistent NAT type: N/A
 Persistent NAT mapping type:  address-port-mapping
 Inactivity timeout:    0
 Max session number:  0
 Translation hits:    0
 Successful sessions:   0
 Failed sessions:   0
 Number of sessions:    0
```

Site-to-Site with Overlapping Subnets

In this case, you have two locations that have overlapping subnets of 11.10.0.0/24 on both sides. To solve that, a host can communicate with a host on the other side. You need to use NAT and target a specific subnet of 11.0.10x.0/24 that translates this request to the remote end. In this case, use *un-numbered* ST interfaces, while you also need to use the `proxy-arp` statement which you'll want if you have a *numbered* ST interface instead.

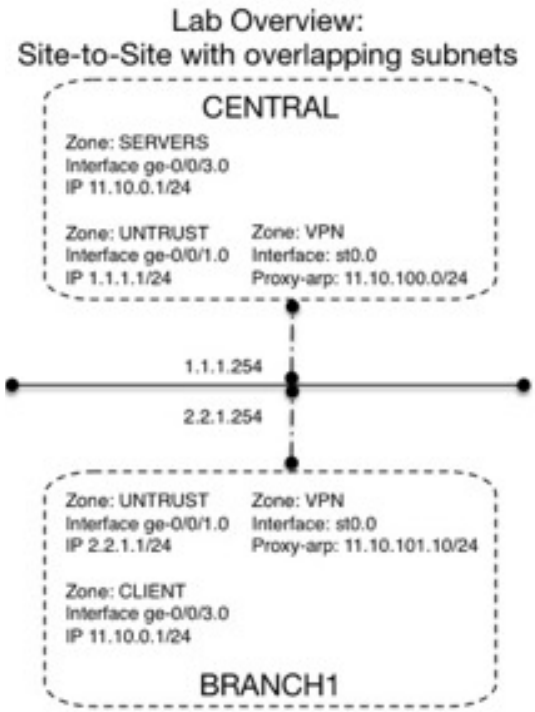


Figure 3.4 *Site-to-Site with Overlapping Subnets*

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used.

Routing: Static

CA: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in

this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site:

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configuration, you must now set a new root administrator password:

root@host# set system root-authentication plain-text-password

```
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet
root@host# set interface st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Configure a route to reach the remote side:

```
root@host# set routing-options static route 11.10.101.0/24 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority

root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the session-init statement for logging to reduce log size.

First build the policy that allows incoming traffic to the CA server from your remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top

root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Somehow you need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:  X
ocsp_rev_status_revoked:  X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells us at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
```

```
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

Now configure a NAT translation of the IPs to reach the other side. If you have a numbered-st-interface, you could skip the `proxy-arp` statement below. Let's use the pool below to hide our internal IPs so the remote end can respond back, without seeing the source IP request as a local broadcast IP:

```
root@CENTRAL# edit security nat
root@CENTRAL# set source pool Branch1 address 11.10.100.0/24
root@CENTRAL# set proxy-arp interface st0.0 address 11.10.100.0/24
root@CENTRAL# edit source rule-set to-branch1
root@CENTRAL# set from zone SERVERS
root@CENTRAL# set to interface st0.0
root@CENTRAL# set rule 1 match source-address 11.10.0.0/24
root@CENTRAL# set rule 1 then source-nat pool Branch1
root@CENTRAL# top
```

Configure a NAT translation, so when the remote site sends traffic to 11.10.100.0/24, it will translate that to 11.10.0.0/24:

```
root@CENTRAL# top edit security nat static rule-set IPSEC
root@CENTRAL# set from zone VPN
root@CENTRAL# set rule 1 match destination-address 11.10.100.0/24
root@CENTRAL# set rule 1 then static-nat prefix 11.10.0.0/24
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured before we can verify our topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet
root@host# set interface st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure a route to reach the remote side:

```
root@host# set routing-options static route 11.10.100.0/24 next-hop st0.0
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing any incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Verify that you trust the certificate by running the this `show` command. You can see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see if you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=Branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@BRANCH1# run show security pki statistics | match ocsp_rev_status
```

```
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Now configure a NAT translation of the IPs to reach the other side. If you have a numbered-st interface, you could skip the `proxy-arp` statement below. Let's use the pool below to hide our internal IPs so the remote end can respond back, without seeing the source IP request as a local broadcast IP:

```
root@BRANCH1# edit security nat
root@BRANCH1# set source pool Branch1 address 11.10.101.0/24
root@BRANCH1# set proxy-arp interface st0.0 address 11.10.101.0/24
root@BRANCH1# edit source rule-set to-branch1
root@BRANCH1# set from zone CLIENTS
root@BRANCH1# set to interface st0.0
root@BRANCH1# set rule 1 match source-address 11.10.0.0/24
root@BRANCH1# set rule 1 then source-nat pool Branch1
root@BRANCH1# top
```

Configure a NAT translation, so when the remote site sends traffic to 11.10.101.0/24 it will translate to 11.10.0.0/24:

```
root@BRANCH1# edit security nat static rule-set IPSEC
root@BRANCH1# set from zone VPN
root@BRANCH1# set rule 1 match destination-address 11.10.101.0/24
root@BRANCH1# set rule 1 then static-nat prefix 11.10.0.0/24
root@BRANCH1# top
```

And add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the config!

```
root@BRANCH1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the Troubleshooting section later in this chapter. Each local site should have a client from which you send traffic if you want to verify that traffic floating through the system, otherwise you need to configure local policies to allow junos-host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 2
 ID        Algorithm    SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  1b294297 1623/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  d5205ec5 1623/ unlim  - root 500  2.2.1.1
```

This should show that your traffic matches the needed NAT policies. Keep in mind that your target is the 11.10.10x.0 address on the remote side:

```
root@CENTRAL# run show security nat static rule all
Total static-nat rules: 1
Total referenced IPv4/IPv6 ip-prefixes: 2/0
Static NAT rule: 1    Rule-set: IPSEC
 Rule-Id:      1
 Rule position:  1
 From zone:  VPN
 Destination addresses: 11.10.100.0
 Host addresses:    11.10.0.0
 Netmask:      24
 Host routing-instance: N/A
 Translation hits:    1
 Successful sessions:   1
 Failed sessions:   0
 Number of sessions:    1
```

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  7204390a 3303/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  53a506ec 3303/ unlim  - root 500  1.1.1.1
```

This should show that your traffic matches the needed NAT policies. Keep in mind that your target is the 11.10.10x.0 address on the remote side:

```
root@BRANCH1# run show security nat static rule all
Total static-nat rules: 1
Total referenced IPv4/IPv6 ip-prefixes: 2/0
Static NAT rule: 1    Rule-set: IPSEC
```

```
Rule-Id:      1
Rule position:  1
From zone:  VPN
Destination addresses: 11.10.101.0
Host addresses:    11.10.0.0
Netmask:      24
Host routing-instance: N/A
Translation hits:    1
Successful sessions:   1
Failed sessions:   0
Number of sessions:    1
```

Site-to-Site with Overlapping Subnets on More Than One Site

In this site-to-site use case, there are two remote locations that have the same subnet of 11.10.101.0/24 and we need to reach both of those locations from the central side (typical from a mananged service). To do this you need to use two fake networks whom hosts from the central side can use as destination networks, in this case 192.168.10.0/24 for Branch1 and 192.168.20.0/24 for Branch2.

Now when a host in the central network targets a destination of 192.168.10.100, it will be mapped to 11.10.101.100 on the Branch1 site. And the opposite if the target is 192.168.20.100.

To do this, you need virtual routers on the central side as the default route table can separate the traffic between two tunnels without load balancing the traffic, which would result in unstable network behavior at some time when you reach one branch, and in some cases the other branch. In this case, we only have traffic from the central side and not any initiated traffic from the spokes – that would require more configuration.

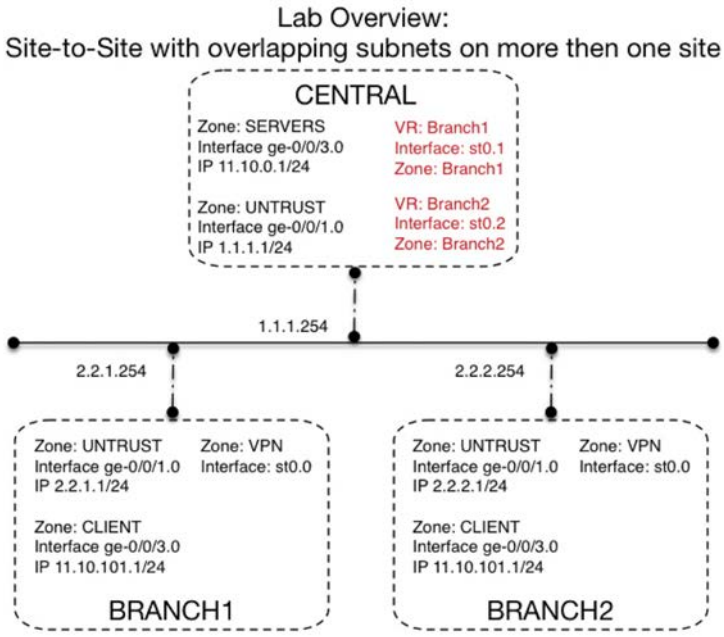


Figure 3.5 *Site-to-Site with Overlapping Subnets on More Than One Site*

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate Authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

root@host# set system root-authentication plain-text-password

```
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description server family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 1 description Site-to-Site-Branch1 family inet mtu 1400
root@host# set interfaces st0 unit 2 description Site-to-Site-Branch2 family inet mtu 1400
root@host# top
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure a security zone for each remote location, assigning the ST interface to the zone. You can have both ST interfaces in one zone, but if you must separate them because most likely it's two different organizations, in that case you want logical separation:

```
root@host# edit security zones security-zone Branch1
root@host# set interfaces st0.1
root@host# top

root@host# edit security zones security-zone Branch2
root@host# set interfaces st0.2
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Now define a routing instance for each branch so you can steer traffic to each branch because they have overlapping subnets:

```
root@host# edit routing-instances Branch1
root@host# set instance-type virtual-router
root@host# set interface st0.1
root@host# set routing-options static route 11.10.101.0/24 next-hop st0.1
root@host# top

root@host# edit routing-instances Branch2
root@host# set instance-type virtual-router
root@host# set interface st0.2
root@host# set routing-options static route 11.10.101.0/24 next-hop st0.2
root@host# top
```

You need to route a fake network for each remote location that the server network's hosts can access to reach the branch networks. This is needed because both branch networks have the same subnet. As you can see, the next-hop used is the previously configured routing instance for each branch. (Later, we will configure static NAT, which will translate the fake network to the original network of the branch site):

```
root@host# set routing-options static route 192.168.10.0/24 next-table Branch1.inet.0
root@host# set routing-options static route 192.168.20.0/24 next-table Branch2.inet.0
```

Time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Here specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top


root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@host# commit
```

Verify that we reach our gateway and also one of the spokes gateway using icmp ping.

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority
root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address CENTRAL-Network 11.10.0.0/24
root@CENTRAL# set address Branch1-Network 11.10.101.0/24
root@CENTRAL# set address Branch2-Network 11.10.101.0/24
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the session-init statement for logging to reduce log size.

First out is to configure the policy to allow incoming traffic to the CA server from the remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for allowing the server network to reach each branch site. It needs to be applied in the global policy and not between the server zone and branch1 or branch2 zones:

```
root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address CENTRAL-Network
root@CENTRAL# set policy 1 match destination-address Branch1-Network
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then permit
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# set policy 2 match source-address CENTRAL-Network
root@CENTRAL# set policy 2 match destination-address Branch2-Network
root@CENTRAL# set policy 2 match application any
root@CENTRAL# set policy 2 then permit
root@CENTRAL# set policy 2 then log session-init session-close
root@CENTRAL# top
```

As it's good to have a clean-up policy, add another global policy that will deny any other traffic and log this. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. But for security precautions, add this:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 254 match source-address any
root@CENTRAL# set policy 254 match destination-address any
root@CENTRAL# set policy 254 match application any
root@CENTRAL# set policy 254 then deny
root@CENTRAL# set policy 254 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to URL: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells us at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top

root@CENTRAL# edit security ike gateway Branch2
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.2.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.1
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top

root@CENTRAL# edit security ipsec vpn Branch2
root@CENTRAL# set bind-interface st0.2
root@CENTRAL# set ike gateway Branch2
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

Define the static NATing that can help us direct traffic to each customer because they have the same networks:

```
root@CENTRAL# edit security nat static rule-set DUPLICATE-IP-NET
root@CENTRAL# set from zone SERVERS
root@CENTRAL# set rule Branch1 match source-address 0.0.0.0/0
root@CENTRAL# set rule Branch1 match destination-address 192.168.10.0/24
root@CENTRAL# set rule Branch1 then static-nat prefix 11.10.101.0/24
root@CENTRAL# set rule Branch1 then static-nat prefix routing-instance Branch1
root@CENTRAL# set rule Branch2 match source-address 0.0.0.0/0
root@CENTRAL# set rule Branch2 match destination-address 192.168.20.0/24
root@CENTRAL# set rule Branch2 then static-nat prefix 11.10.101.0/24
root@CENTRAL# set rule Branch2 then static-nat prefix routing-instance Branch2
root@CENTRAL# top
```

Add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured so you can verify the topology:

```
root@CENTRAL# commit
```

Configure the Spoke

Before you begin, note that configuration pieces displayed in boldface should be changed for the second branch: Branch2.

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming ssh access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss):
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces st0 unit 1 description Site-to-Site-VPN family inet mtu 1400
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing any incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
```

```
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.1
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure a route to find the central side:

```
root@host# set routing-options static route 11.10.0.0/24 next-hop st0.1
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

NOTE From now on, the host will not have the same name in your syntax on the second machine. Keep in mind it will have the hostname you used at the top of this section.

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address CENTRAL-Network 11.10.0.0/24
root@BRANCH1# set security address-book global address BRANCH-Network 11.10.101.0/24
```

Now define a policy between the VPN zone and the CLIENT zone, on the branch side, but do not specify the polices under the global zone:

```
root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address CENTRAL-Network
root@BRANCH1# set match destination-address BRANCH-Network
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

By running is command, you can see that if you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service.

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate.

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running is command, you can see that if you have a successful verification or not. Again, if the number is not updated then you have a problem with the certificate or the OCSP response service.

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Stockholm,O=Myd
omain,OU=LAB,CN=Branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

Configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name.

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.1
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
```

```
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

And commit the configuration.

```
root@BRANCH1# commit
```

Verification

Time to verify that your configuration is working. If something is not working, go to the Troubleshooting section later in this chapter under site-to-site. Each local site should have a client from which you send traffic if you want to verify that traffic is floating through the system, otherwise you need to configure local policies to allow the Junos host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 2
 ID        Algorithm    SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  1b294297 1623/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  d5205ec5 1623/ unlim  - root 500  2.2.1.1
 <131074   ESP:aes-cbc-256/sha256  c4f79c35 3244/ unlim  - root 500  2.2.2.1
 >131074   ESP:aes-cbc-256/sha256  7f7fa508 3244/ unlim  - root 500  2.2.2.1
```

This will show you what active routes you have installed; you should see all of the routes displayed for this solution to work:

```
root@CENTRAL# run show route | match st
inet.0: 8 destinations, 9 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
0.0.0.0/0  *[Static/5] 1d 17:38:52
192.168.10.0/24  *[Static/5] 20:59:27
192.168.20.0/24  *[Static/5] 20:59:27

Branch1.inet.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.101.0/24 *[Static/5] 20:30:06
> via st0.1

Branch2.inet.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
11.10.101.0/24 *[Static/5] 20:04:37
> via st0.2
```

Now let's verify that the traffic hits the NAT policies:

```
root@CENTRAL# run show security nat static rule all
Total static-nat rules: 2
Total referenced IPv4/IPv6 ip-prefixes: 4/0

Static NAT rule: Branch1Rule-set: DUPLICATE-IP-NET
 Rule-Id:      1
 Rule position:  1
 From zone:  SERVERS
 Source addresses:    0.0.0.0 - 255.255.255.255
 Destination addresses: 192.168.10.0
 Host addresses:    11.10.101.0
 Netmask:      24
 Host routing-instance: Branch1
 Translation hits:    1
 Successful sessions:   1
 Failed sessions:   0
 Number of sessions:    1

Static NAT rule: Branch2Rule-set: DUPLICATE-IP-NET
 Rule-Id:      2
 Rule position:  2
 From zone:  SERVERS
 Source addresses:    0.0.0.0 - 255.255.255.255
 Destination addresses: 192.168.20.0
 Host addresses:    11.10.101.0
 Netmask:      24
 Host routing-instance: Branch2
 Translation hits:    0
 Successful sessions:   0
 Failed sessions:   0
 Number of sessions:    0
```

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm    SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  c97a0cf1   1509/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  35d641d5   1509/ unlim  - root 500  1.1.1.1

And that you have a route to the target destination:
root@BRANCH1# run show route | match st
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
0.0.0.0/0    *[Static/5] 1d 20:12:32
11.10.0.0/24 *[Static/5] 23:45:26
> via st0.1
```

Because there isn't any NAT on this side, only requesting the remote side, it's less information to verify.

Site-to-Site with OSPF

In this use case, there are two locations that want to use OSPF as the routing protocol between the networks over IPsec. Keep in mind that SRX configures the ST interface as a P2P interface by default, which is why this should be changed to P2MP, if you plan to use a hub-and-spoke topology.

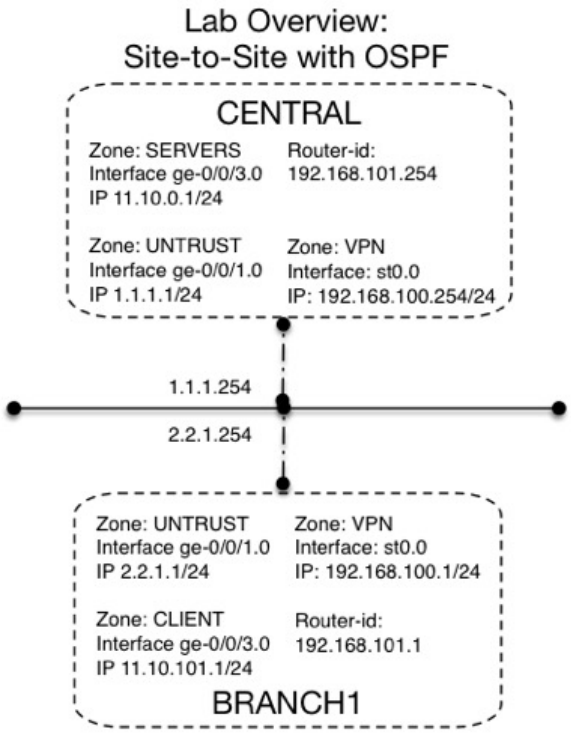


Figure 3.6 *Site-to-Site with OSPF*

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate Authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but

those are not described in this cookbook. Keep in mind that this cookbook has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet address
192.168.100.254/24
root@host# set interface st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Configure OSPF and a router ID:

```
root@host# set routing-options router-id 192.168.101.254
root@host# set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 passive
root@host# set protocols ospf area 0.0.0.0 interface st0.0 metric 1000
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.0 host-inbound-traffic protocol ospf
root@host# top
```

Time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Here specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# set revocation-check use-ocsp
root@host# top
```

Keep in mind that the challenge-password needs to match your CA servers. Here is an example:

```
root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
```

```
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to commit and verify your IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority

root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the session-init statement for logging to reduce log size.

First create the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top

root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

You need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all
root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@CENTRAL# run request security pki generate-key-pair size 2048 type rsa certificate-id CENTRAL
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells us at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured so you can verify the topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming ssh access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces st0 unit 0 description Site-to-Site-VPN family inet address 192.168.100.1/24
root@host# set interfaces st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure OSPF and a router ID:

```
root@host# set routing-options router-id 192.168.101.1
root@host# set protocols ospf area 0.0.0.0 interface ge-0/0/3.0 passive
root@host# set protocols ospf area 0.0.0.0 interface st0.0 metric 1000
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing any incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.0 host-inbound-traffic protocol ospf
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service.

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair size 2048 type rsa certificate-id BRANCH1
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=branch1 challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@BRANCH1L# run request security pki local-certificate verify certificate-id BRANCH1

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
```

```
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the configuration:

```
root@BRANCH1# commit
```

Verification

Verify that your configuration is working. If something is not working, go to the troubleshooting section under site-to-site. Each local site should have a client from which you send traffic if you want to verify that traffic floating through the system, otherwise you need to configure local policies to allow junos-host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  c7ad4208 3242/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  9e9e0b6  3242/ unlim  - root 500  2.2.1.1
```

This shows that your OSPF peering is working and what prefixes are exchanged:

```
root@CENTRAL# run show route protocol ospf
inet.0: 10 destinations, 12 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.101.0/24 *[OSPF/10] 00:04:23, metric 1001
> via st0.0
192.168.100.0/24 [OSPF/10] 00:04:23, metric 1000
> via st0.0
224.0.0.5/32 *[OSPF/10] 00:04:38, metric 1
MultiRecv
```

NOTE If you don't see your routes, you most likely have a OSPF misconfiguration.

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  9e9e0b6  3297/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  c7ad4208 3297/ unlim  - root 500  1.1.1.1
```

This shows that your OSPF peering is working and what prefixes are exchanged:

```
root@BRANCH1# run show route protocol ospf
inet.0: 9 destinations, 10 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24 *[OSPF/10] 00:08:20, metric 1001
> via st0.0
192.168.100.0/24 [OSPF/10] 00:08:20, metric 1000
> via st0.0
224.0.0.5/32 *[OSPF/10] 00:09:04, metric 1
MultiRecv
```

NOTE If you don't see your routes, you most likely have an OSPF misconfiguration.

Site-to-Site with BGP

In this case, there are two locations that want to use BGP as the routing protocol between the networks over IPsec. Keep in mind that SRX configures the ST interface as a P2P interface by default, which is why this should be changed to P2MP if you plan to use a hub-and-spoke topology.

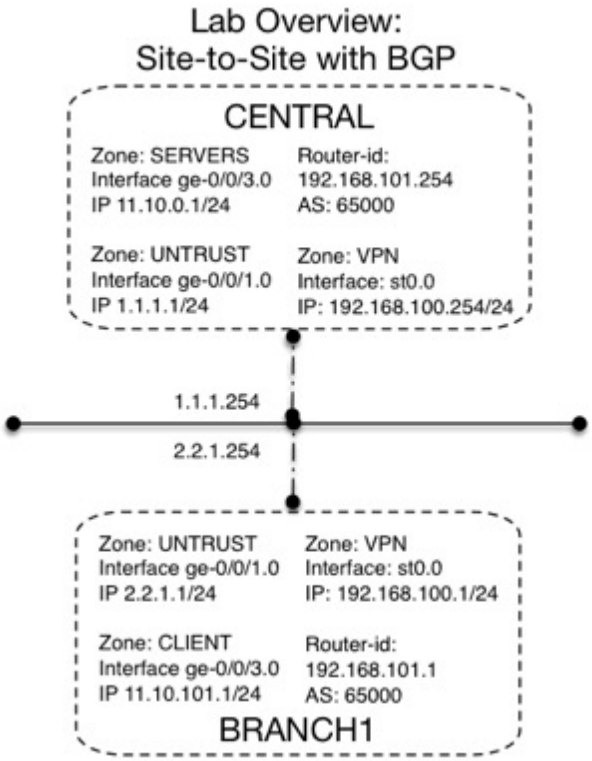


Figure 3.7 Site-to-Site with BGP

Requirements

Hardware: Juniper SRX Service Gateways

Software: Junos 12.3X48D10 is used

Routing: Static

Certificate Authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but

those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name CENTRAL
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# edit interface
root@host# set ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set st0 unit 0 description Site-to-Site-VPN family inet address 192.168.100.254/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Configure BGP and a router ID:

```
root@host# set routing-options router-id 192.168.101.254
root@host# set routing-options autonomous-system 6500
root@host# edit protocols bgp group IBGP
root@host# set type internal
root@host# set import FROM_BGP
root@host# set export INTO_BGP
root@host# set neighbor 192.168.100.1
root@host# top
```

You now have to define and control what routes that should be imported and exported from BGP. The next active configuration will allow all static and direct routes that are active to be advertised into BGP, and all routes from BGP to be redistributed. This can be controlled by activating the prefix-lists statements and adding the routes that you want to export/import into the added prefix-lists below:

```
root@host# edit policy-options policy-statement FROM_BGP
root@host# set term FROM_BGP from protocol bgp
root@host# set term FROM_BGP from prefix-list FROM_BGP
root@host# deactivate term FROM_BGP from prefix-list FROM_BGP
root@host# set term FROM_BGP then accept
root@host# set term DENY then reject
root@host# top

root@host# edit policy-options policy-statement INTO_BGP
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT from prefix-list DIRECT_INTO_BGP
root@host# set term DIRECT from state active
root@host# set term DIRECT then next-hop self
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC from prefix-list STATIC_INTO_BGP
root@host# deactivate term STATIC from prefix-list STATIC_INTO_BGP
root@host# set term STATIC from state active
root@host# set term STATIC then next-hop self
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top
```

As our external interface would match protocol direct, you need to define which connected interfaces should be advertised with BGP (this can also be done in other ways). Here we use a prefix list to control this behavior:

```
root@host# set policy-options prefix-list DIRECT_INTO_BGP 11.10.0.0/24
root@host# set policy-options prefix-list STATIC_INTO_BGP
root@host# set policy-options prefix-list FROM_BGP
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.0 host-inbound-traffic protocols bgp
root@host# top
```

Now it's time to configure and request the certificates that you need to establish the tunnel. First, define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Here specify the CA SCEP URL:

```
root@host# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@host# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top
```

The challenge password needs to match your CA server. This is just an example:

```
root@host# edit security pki auto-re-enrollment certificate-id CENTRAL
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password 6C3CC94AE7BDA240F110122F0612BDB2
root@host# set scep-encryption-algorithm des3
root@host# top
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@host# commit
```

Verify the gateway using an ICMP ping:

```
root@CENTRAL# run ping 1.1.1.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@CENTRAL# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Now configure the Destination NAT rule configuration:

```
root@CENTRAL# edit security nat destination
root@CENTRAL# set pool CertificateAuthority address 11.10.0.10/32
root@CENTRAL# edit rule-set CertificateAuthority
root@CENTRAL# set from interface ge-0/0/1.0
root@CENTRAL# set rule 1 match source-address 0.0.0.0/0
root@CENTRAL# set rule 1 match destination-address 1.1.1.2/32
root@CENTRAL# set rule 1 then destination-nat pool CertificateAuthority
root@CENTRAL# top
```

In this step you configure the different address book objects that you will use later on to build your security policies:

```
root@CENTRAL# edit security address-book global
root@CENTRAL# set address DATA-Networks 11.10.0.0/16
root@CENTRAL# set address CertificateAuthority 11.10.0.10/32
root@CENTRAL# top
```

Here you start building the security policy that will define what traffic that should be allowed in different directions.

NOTE In a production environment, you can remove the `session-init` statement for logging to reduce log size).

First build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@CENTRAL# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@CENTRAL# set match source-address any
root@CENTRAL# set match destination-address CertificateAuthority
root@CENTRAL# set match application junos-http
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

Now define a policy for each direction between SERVERS and VPN:

```
root@CENTRAL# edit security policies from-zone SERVERS to-zone VPN policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

```
root@CENTRAL# edit security policies from-zone VPN to-zone SERVERS policy 1
root@CENTRAL# set match source-address DATA-Networks
root@CENTRAL# set match destination-address DATA-Networks
root@CENTRAL# set match application any
root@CENTRAL# set then permit
root@CENTRAL# set then log session-init session-close
root@CENTRAL# top
```

You need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@CENTRAL# set security policies default-policy deny-all

root@CENTRAL# edit security policies global
root@CENTRAL# set policy 1 match source-address any
root@CENTRAL# set policy 1 match destination-address any
root@CENTRAL# set policy 1 match application any
root@CENTRAL# set policy 1 then deny
root@CENTRAL# set policy 1 then log session-init session-close
root@CENTRAL# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@CENTRAL# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:  X
ocsp_rev_status_revoked:  X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@CENTRAL# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@CENTRAL# run request security pki generate-key-pair certificate-id CENTRAL size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. This is explained under the syntax:

```
root@CENTRAL# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
CENTRAL domain-name central.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O=Myd
omain,OU=LAB,CN=Central challenge-password 6C3CC94AE7BDA240F110122F0612BDB2
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@CENTRAL# run request security pki local-certificate verify certificate-id CENTRAL

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@CENTRAL# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@CENTRAL# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set authentication-method rsa-signatures
root@CENTRAL# set dh-group group14
root@CENTRAL# set authentication-algorithm sha-256
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 28800
root@CENTRAL# top
```

IKE Policy

The name tells us at a glance that it's used for Site-to-Site VPNs. Let's bind our `local-certificate` to this policy:

```
root@CENTRAL# edit security ike policy Site-to-Site
root@CENTRAL# set mode main
root@CENTRAL# set proposals CERT-DH14-SHA256-AES256-L28800
root@CENTRAL# set certificate local-certificate CENTRAL
root@CENTRAL# set certificate peer-certificate-type x509-signature
root@CENTRAL# top
```

IKE Gateway

Now configure how to authenticate the spokes that you want to establish a Site-to-Site tunnel. Here it's `Branch1` because that is the lab's remote peer. This will then be bound together in the IPsec VPN configuration:

```
root@CENTRAL# edit security ike gateway Branch1
root@CENTRAL# set ike-policy Site-to-Site
root@CENTRAL# set dead-peer-detection probe-idle-tunnel
root@CENTRAL# set remote-identity distinguished-name
root@CENTRAL# set local-identity distinguished-name
root@CENTRAL# set external-interface ge-0/0/1.0
root@CENTRAL# set address 2.2.1.1
root@CENTRAL# set version v2-only
root@CENTRAL# top
```

IPsec Proposal

Here is the proposal parameter used in IPsec policies:

```
root@CENTRAL# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@CENTRAL# set protocol esp
root@CENTRAL# set authentication-algorithm hmac-sha-256-128
root@CENTRAL# set encryption-algorithm aes-256-cbc
root@CENTRAL# set lifetime-seconds 3600
root@CENTRAL# top
```

IPsec Policy

The IPsec Policy binds the IPsec Proposals to be used in the IPsec VPN configuration:

```
root@CENTRAL# edit security ipsec policy Site-to-Site
root@CENTRAL# set perfect-forward-secrecy keys group14
root@CENTRAL# set proposals ESP-SHA256-AES256-L3600
root@CENTRAL# top
```

IPsec VPN

Bind together the IKE Gateway and IPsec policy and the Secure Tunnel Interface:

```
root@CENTRAL# edit security ipsec vpn Branch1
root@CENTRAL# set bind-interface st0.0
root@CENTRAL# set ike gateway Branch1
root@CENTRAL# set ike ipsec-policy Site-to-Site
root@CENTRAL# set ike proxy-identity local 0.0.0.0/0
root@CENTRAL# set ike proxy-identity remote 0.0.0.0/0
root@CENTRAL# set establish-tunnels immediately
root@CENTRAL# top
```

And add syslog for troubleshooting:

```
root@CENTRAL# set system syslog user * any emergency
root@CENTRAL# edit system syslog
root@CENTRAL# set file messages any any
root@CENTRAL# set file messages authorization info
root@CENTRAL# set file messages change-log none
root@CENTRAL# set file messages interactive-commands none
root@CENTRAL# set file messages structured-data
root@CENTRAL# set file interactive-commands interactive-commands any
root@CENTRAL# top
```

It's time to commit and wait for the other spoke to be configured so you can verify the topology:

```
root@CENTRAL# commit
```

Configure the Spoke

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name BRANCH1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces:

```
root@host# edit interface
root@host# set ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set st0 unit 0 description Site-to-Site-VPN family inet address 192.168.100.1/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure BGP and a router ID together with an autonomous system ID and neighbor:

```
root@host# set routing-options autonomous-system 6500
root@host# set routing-options router-id 192.168.101.1
root@host# edit protocols bgp group IBGP
root@host# set type internal
root@host# set import FROM_BGP
root@host# set export INTO_BGP
root@host# set neighbor 192.168.100.254
root@host# top
```

Now define and control what routes should be imported and exported to and from BGP.

The next active config will allow all static and direct routes that are active to be advertised into BGP, and all routes from BGP to be re-distributed. This can be controlled by activating the prefix-lists statements below and adding the routes that you want to export or import into the added prefix-lists:

```
root@host# edit policy-options policy-statement FROM_BGP
root@host# set term FROM_BGP from protocol bgp
root@host# set term FROM_BGP from prefix-list FROM_BGP
root@host# deactivate term FROM_BGP from prefix-list FROM_BGP
root@host# set term FROM_BGP then accept
root@host# set term DENY then reject
root@host# top
```

```
root@host# edit policy-options policy-statement INTO_BGP
root@host# set term DIRECT from protocol direct
root@host# set term DIRECT from prefix-list DIRECT_INTO_BGP
root@host# set term DIRECT from state active
root@host# set term DIRECT then next-hop self
root@host# set term DIRECT then accept
root@host# set term STATIC from protocol static
root@host# set term STATIC from prefix-list STATIC_INTO_BGP
root@host# deactivate term STATIC from prefix-list STATIC_INTO_BGP
root@host# set term STATIC from state active
root@host# set term STATIC then next-hop self
root@host# set term STATIC then accept
root@host# set term DENY then reject
root@host# top
```

As our external interface would match protocol direct, you need to define which connected interfaces should be advertised with BGP (this can also be done in other ways). Here, we used a prefix list to control this behavior:

```
root@host# set policy-options prefix-list DIRECT_INTO_BGP 11.10.101.0/24
root@host# set policy-options prefix-list STATIC_INTO_BGP
root@host# set policy-options prefix-list FROM_BGP
```

Configure the security zone UNTRUST, assigning the interface to the zone, plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone CLIENTS, assigning the interface to the zone, and allowing any incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
root@host# top
```

Configure the security zone VPN, assigning the interface to the zone plus allowing any incoming administrative services:

```
root@host# edit security zones security-zone VPN
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.0 host-inbound-traffic protocols bgp
root@host# top
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@host# edit security pki ca-profile Our-CA_Server
root@host# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@host# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Here we configure how to verify the validity of the certificate, we have disabled the verification in this step:

```
root@host# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@host# set revocation-check use-ocsp ocsp nonce-payload enable
root@host# top

root@host# edit security pki auto-re-enrollment certificate-id BRANCH1
root@host# set ca-profile-name Our-CA_Server
root@host# set re-generate-keypair
root@host# set re-enroll-trigger-time-percentage 10
root@host# set challenge-password "$9$Vxw4aGDi"
root@host# set scep-encryption-algorithm des3
root@host# top
```

It's time to activate this configuration and then verify IP connectivity before continuing:

```
root@host# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@BRANCH1# run ping 2.2.1.254
root@BRANCH1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@BRANCH1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Define a policy for each direction between CLIENTS and VPN:

```
root@BRANCH1# edit security policies from-zone CLIENTS to-zone VPN policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top

root@BRANCH1# edit security policies from-zone VPN to-zone CLIENTS policy 1
root@BRANCH1# set match source-address DATA-Networks
root@BRANCH1# set match destination-address DATA-Networks
root@BRANCH1# set match application any
root@BRANCH1# set then permit
root@BRANCH1# set then log session-init session-close
root@BRANCH1# top
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@BRANCH1# set security policies default-policy deny-all

root@BRANCH1# edit security policies global
root@BRANCH1# set policy 1 match source-address any
root@BRANCH1# set policy 1 match destination-address any
root@BRANCH1# set policy 1 match application any
root@BRANCH1# set policy 1 then deny
root@BRANCH1# set policy 1 then log session-init session-close
root@BRANCH1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@BRANCH1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Verify that you trust the certificate:

```
root@BRANCH1# run request security pki ca-certificate verify ca-profile Our-CA_Server
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@BRANCH1# run request security pki generate-key-pair certificate-id BRANCH1 size 2048 type rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@BRANCH1# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
BRANCH1 domain-name branch1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.com,L=Oslo,O=Mydomain
,OU=LAB,CN=Branch1 challenge-password 6C3CC94AE7BDA240F110122F0612BDB2
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

```
root@BRANCH1# run request security pki local-certificate verify certificate-id BRANCH1
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@BRANCH1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@BRANCH1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set authentication-method rsa-signatures
root@BRANCH1# set dh-group group14
root@BRANCH1# set authentication-algorithm sha-256
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 28800
root@BRANCH1# top
```

IKE Policy

In this example we use *Site-to-Site* as the name:

```
root@BRANCH1# edit security ike policy Site-to-Site
root@BRANCH1# set mode main
root@BRANCH1# set proposals CERT-DH14-SHA256-AES256-L28800
root@BRANCH1# set certificate local-certificate BRANCH1
root@BRANCH1# set certificate peer-certificate-type x509-signature
root@BRANCH1# top
```

IKE Gateway

```
root@BRANCH1# edit security ike gateway CENTRAL
root@BRANCH1# set ike-policy Site-to-Site
root@BRANCH1# set address 1.1.1.1
root@BRANCH1# set dead-peer-detection probe-idle-tunnel
root@BRANCH1# set local-identity distinguished-name
root@BRANCH1# set remote-identity distinguished-name
root@BRANCH1# set external-interface ge-0/0/1.0
root@BRANCH1# set version v2-only
root@BRANCH1# top
```

IPsec Proposal

```
root@BRANCH1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@BRANCH1# set protocol esp
root@BRANCH1# set authentication-algorithm hmac-sha-256-128
root@BRANCH1# set encryption-algorithm aes-256-cbc
root@BRANCH1# set lifetime-seconds 3600
root@BRANCH1# top
```

IPsec Policy

```
root@BRANCH1# edit security ipsec policy Site-to-Site
root@BRANCH1# set perfect-forward-secrecy keys group14
root@BRANCH1# set proposals ESP-SHA256-AES256-L3600
root@BRANCH1# top
```

IPsec VPN

```
root@BRANCH1# edit security ipsec vpn CENTRAL
root@BRANCH1# set bind-interface st0.0
root@BRANCH1# set ike gateway CENTRAL
root@BRANCH1# set ike ipsec-policy Site-to-Site
root@BRANCH1# set ike proxy-identity local 0.0.0.0/0
root@BRANCH1# set ike proxy-identity remote 0.0.0.0/0
root@BRANCH1# set establish-tunnels immediately
root@BRANCH1# top
```

Add syslog for troubleshooting:

```
root@BRANCH1# set system syslog user * any emergency
root@BRANCH1# edit system syslog
root@BRANCH1# set file messages any any
root@BRANCH1# set file messages authorization info
root@BRANCH1# set file messages change-log none
root@BRANCH1# set file messages interactive-commands none
root@BRANCH1# set file messages structured-data
root@BRANCH1# set file interactive-commands interactive-commands any
root@BRANCH1# top
```

Commit the configuration:

```
root@BRANCH1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the troubleshooting section under site-to-site. Each local site should have a client from which you send traffic if you want to verify that traffic is floating through the system, otherwise you need to configure local policies to allow the Junos host to send traffic between certain zones.

On Central

This shows that your configured IPsec tunnels are up and active:

```
root@CENTRAL# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  c7ad4208 3242/ unlim  - root 500  2.2.1.1
 >131073   ESP:aes-cbc-256/sha256  9e9e0b6  3242/ unlim  - root 500  2.2.1.1
```

This will show that your BGP peering is working and what prefixes are exchanged:

```
root@CENTRAL# run show route protocol bgp
inet.0: 9 destinations, 10 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.101.0/24 *[BGP/170] 00:04:59, localpref 100
AS path: I, validation-state: unverified
> to 192.168.100.1 via st0.0
```

NOTE If you don't see your routes, you most likely have a BGP misconfiguration.

On Spoke

This shows that your configured IPsec tunnels are up and active:

```
root@BRANCH1# run show security ipsec security-associations
 Total active tunnels: 1
 ID        Algorithm      SPI  Life:sec/kb    Mon  lsys Port Gateway
 <131073   ESP:aes-cbc-256/sha256  9e9e0b6  3297/ unlim  - root 500  1.1.1.1
 >131073   ESP:aes-cbc-256/sha256  c7ad4208 3297/ unlim  - root 500  1.1.1.1
```

This will show that your BGP peering is working and what prefixes it is exchanging:

```
root@BRANCH1# run show route protocol bgp
inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24 *[BGP/170] 00:04:41, localpref 100
AS path: I, validation-state: unverified
> to 192.168.100.254 via st0.0
```

NOTE If you don't see your routes, you most likely have a BGP misconfiguration.

Chapter 4

Configure ADVPN or GroupVPNv2

The challenge with site-to-site IPsec solutions configured for hub-and-spoke traffic is that all of them will saturate the bandwidth on the hub site when traffic is established between different remote locations. That's because all that data will flow through the hub. With a saturated link, you will also experience latency and jitter impacting collaboration services and other latency and jitter sensitive services. Services that require more bandwidth can be impacted as well.

The solution to these challenges is Auto Discovery VPN (ADVPN) and it is deployed in conjunction with AutoVPN.

ADVPN always has a central hub and two or more spokes; the hub is referred to as the "*suggester*" and each spoke as a "*partner*." On the suggester side, you can see that each partner has established an IPsec tunnel. This means that you now have a partner established in the topology. When you verify Security Associations (SAs) you define this tunnel as static and that the hub device is in *suggester* mode; if you check the partner device it is in *partner* mode.

For the suggester to be able to notify the partners to set up dynamic tunnels, you need to use OSPF between the suggester and each partner. You will have an OSPF neighbor between the suggester and the partner so the suggester can send exchange updates to its partners that are starting to exchange data; as a result, one of the partners will try to establish a dynamic tunnel between the two partners.

NOTE Currently Juniper does not support SRX HE as a partner but it is planned soon after this cookbook is published.

NOTE If spoke1 already has a shortcut with spoke2 and loses its communication with the hub site, what happens when spoke4 wants to exchange data with spoke1? The hub site knows that the shortest path is via spoke2 through the OSPF database; the hub site sends a shortcut exchange to spoke2 and spoke4 that they should establish a shortcut tunnel, as the shortest routing path is via spoke2. Take this into consideration when building your topology. Why? In most cases, either end of the tunnel (spoke1 or hub site) losing its connectivity shouldn't be a big problem. Don't forget to configure the partner connection-limit option as this could easily consume all of your tunnels or make the OSPF database unnecessary large.

In each security zone, you can configure if the IP address belonging to the interface should respond to ping and SSH. If you don't want this, these statements can be removed.

IMPORTANT Do remember that you need at least one SSH-enabled port to access and manage the box:

```
"set security zone security-zone <zone_name> host-inbound-traffic system-services <function/service>"
```

ADVPN with No Redundant Internet Connection

In this ADVPN use case, there are two remote offices that have IPsec tunnels to the headquarters but also shortcut tunnels between the remote offices to offload headquarters bandwidth and improve quality of experience for the users exchanging data. Each site has two segments. None of the sites have redundant Internet connections.

In Figure 4.1 you can only see two spokes but you can add more in the same way as described for the two spokes. Keep in mind that you might need to change the `connection-limit` under each partner if you increase the network to a large extent.

NOTE The configuration is displayed in snippets below. The configuration files are available on this book's landing page at http://www.juniper.net/dayone. Keep in mind that you need to change variable data such as interface, IP addresses, and other information related to your own network.

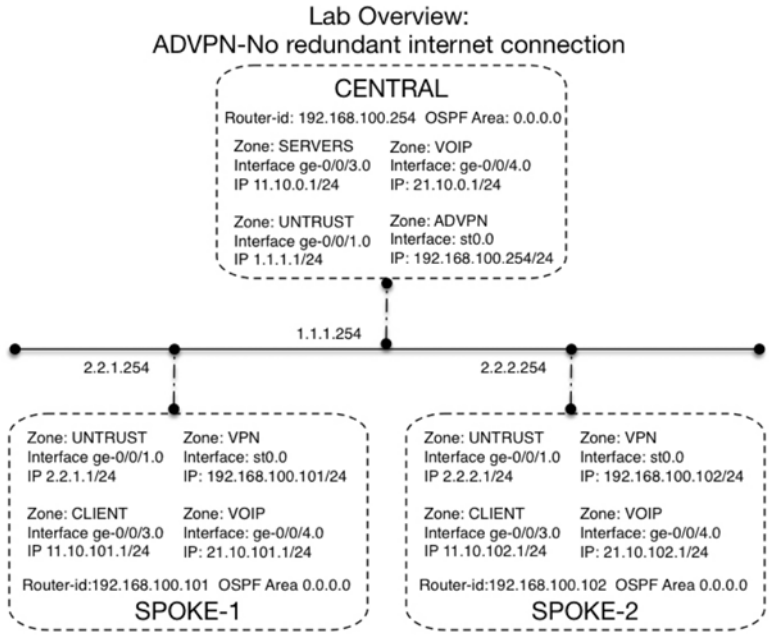


Figure 4.1 ADVPN with No Redundant Internet Connection

Requirements

Hardware: Juniper SRX Series Service Gateway SRX1xx – 650 for Spoke and Suggester deployment, SRX1K to SRX5K can only act as Suggester.

Software: Junos 12.3X48D10 and above

Routing: OSPF in the VPN topology

Certificate authority: SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-HUB
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces, and set the MTU on the ST interface to 1400 because we want to get fragmentation issues on the VPN:

```
root@host# edit interface
root@host# set ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set ge-0/0/4 unit 0 description voip family inet address 21.10.0.1/24
root@host# set st0 unit 0 description advpn multipoint family inet address 192.168.100.254/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.254
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# top edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# top edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# top edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# top
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the configuration:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-HUB# set routing-options router-id 192.168.100.254
```

Configure OSPF:

```
root@ADVPN-HUB# top edit protocols ospf area 0.0.0.0
root@ADVPN-HUB# set interface st0.0 interface-type p2mp
root@ADVPN-HUB# set interface st0.0 metric 10
root@ADVPN-HUB# set interface st0.0 demand-circuit
root@ADVPN-HUB# set interface st0.0 flood-reduction
root@ADVPN-HUB# set interface st0.0 dynamic-neighbors
root@ADVPN-HUB# set interface st0.0 retransmit-interval 1
root@ADVPN-HUB# set interface st0.0 dead-interval 40
root@ADVPN-HUB# set interface ge-0/0/3.0 passive
root@ADVPN-HUB# set interface ge-0/0/4.0 passive
```

Configure and request the certificates that you need to establish the tunnel. First you need to define how to find the CA:

```
root@ADVPN-HUB# top edit security pki ca-profile Our-CA_Server
root@ADVPN-HUB# set ca-identity MydomainCertificateAuthority
```

Specify the CA SCEP URL:

```
root@ADVPN-HUB# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@ADVPN-HUB# set revocation-check use-ocsp ocsp url http://11.10.0.10/ocsp
root@ADVPN-HUB# set revocation-check use-ocsp ocsp nonce-payload enable
```

Keep in mind that the challenge-password is unique to your CA :

```
root@ADVPN-HUB# top edit security pki auto-re-enrollment certificate-id ADVPN-HUB
root@ADVPN-HUB# set ca-profile-name Our-CA_Server
root@ADVPN-HUB# set re-generate-keypair
root@ADVPN-HUB# set re-enroll-trigger-time-percentage 10
root@ADVPN-HUB# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN-HUB# set scep-encryption-algorithm des3
root@ADVPN-HUB# top
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@ADVPN-HUB# commit
```

Verify the gateway using an ICMP ping:

```
root@ADVPN-HUB# run ping 1.1.1.254
root@ADVPN-HUB# run ping 2.2.1.254
root@ADVPN-HUB# run ping 2.2.2.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for Certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@ADVPN-HUB# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Configure the Destination NAT rule configuration:

```
root@ADVPN-HUB# edit security nat destination
root@ADVPN-HUB# set pool CertificateAuthority address 11.10.0.10/32
root@ADVPN-HUB# edit rule-set CertificateAuthority
root@ADVPN-HUB# set from interface ge-0/0/1.0
root@ADVPN-HUB# set rule 1 match source-address 0.0.0.0/0
root@ADVPN-HUB# set rule 1 match destination-address 1.1.1.2/32
root@ADVPN-HUB# set rule 1 then destination-nat pool CertificateAuthority
```

Configure the different address book objects that you will use later on to build your security policies:

```
root@ADVPN-HUB# top edit security address-book global
root@ADVPN-HUB# set address VOIP-Networks 21.20.0.0/16
root@ADVPN-HUB# set address DATA-Networks 11.10.0.0/16
root@ADVPN-HUB# set address CertificateAuthority 11.10.0.10/32
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the `session-init` statement for logging to reduce log size.

First, build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@ADVPN-HUB# top edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address CertificateAuthority
root@ADVPN-HUB# set match application junos-http
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Define a policy for each direction between CLIENTS and ADVPN:

```
root@ADVPN-HUB# top edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

```
root@ADVPN-HUB# top edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Now define a policy for each direction between SERVERS and VPN:

```
root@ADVPN-HUB# top edit security policies from-zone SERVERS to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# top edit security policies from-zone ADVPN to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# top edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address any
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

You need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@ADVPN-HUB# set security policies default-policy deny-all

root@ADVPN-HUB# edit security policies global
root@ADVPN-HUB# set policy 1 match source-address any
root@ADVPN-HUB# set policy 1 match destination-address any
root@ADVPN-HUB# set policy 1 match application any
root@ADVPN-HUB# set policy 1 then deny
root@ADVPN-HUB# set policy 1 then log session-init session-close
root@ADVPN-HUB# top
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@ADVPN-HUB# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@ADVPN-HUB# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@ADVPN-HUB# run request security pki generate-key-pair certificate-id ADVPN-HUB size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-HUB# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
ADVPN-HUB domain-name advpn-hub.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O
=Mydomain,OU=LAB,CN=advpn-hub challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to the URL: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
```

```
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-HUB# run request security pki local-certificate verify certificate-id ADVPN-HUB

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-HUB# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set authentication-method rsa-signatures
root@ADVPN-HUB# set dh-group group14
root@ADVPN-HUB# set authentication-algorithm sha-256
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 28800
```

IKE Policy

The name indicates that it's used for ADVPN topologies, and this device should only serve as a Suggester (hub). Lets bind the local-certificate to this policy:

```
root@ADVPN-HUB# top edit security ike policy ADVPN
root@ADVPN-HUB# set mode main
root@ADVPN-HUB# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set certificate local-certificate ADVPN-HUB
root@ADVPN-HUB# set certificate peer-certificate-type x509-signature
```

IKE Gateway

Configure how to authenticate the spokes that you want to establish an ADVPN tunnel to. Here, named `ADVPN-SPOKES` for the remote peers. The `wildcard OU` name, is what to search for in the remote peer's certificate to authenticate the peer. The IKE gateway will be bound together in the IPsec VPN configuration:

```
root@ADVPN-HUB# top edit security ike gateway ADVPN-SPOKES
root@ADVPN-HUB# set ike-policy ADVPN
root@ADVPN-HUB# set dynamic distinguished-name wildcard OU=LAB
```

```
root@ADVPN-HUB# set dynamic ike-user-type group-ike-id
root@ADVPN-HUB# set local-identity distinguished-name
root@ADVPN-HUB# set external-interface ge-0/0/1.0
root@ADVPN-HUB# set advpn partner disable
root@ADVPN-HUB# set version v2-only
```

IPsec Proposal

Here are the parameters used in the IPsec policies:

```
root@ADVPN-HUB# top edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-HUB# set protocol esp
root@ADVPN-HUB# set authentication-algorithm hmac-sha-256-128
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 3600
```

IPsec Policy

The IPsec policy binds IPsec proposals to be used in the IPsec VPN configuration:

```
root@ADVPN-HUB# top edit security ipsec policy ADVPN
root@ADVPN-HUB# set perfect-forward-secrecy keys group14
root@ADVPN-HUB# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

Now bind together the IKE Gateway and the IPsec Policy with the secure tunnel interface:

```
root@ADVPN-HUB# top edit security ipsec vpn ADVPN-SPOKES
root@ADVPN-HUB# set bind-interface st0.0
root@ADVPN-HUB# set ike gateway ADVPN-SPOKES
root@ADVPN-HUB# set ike ipsec-policy ADVPN
root@ADVPN-HUB# top
```

Add syslog for troubleshooting:

```
root@ADVPN-HUB# set system syslog user * any emergency
root@ADVPN-HUB# edit system syslog
root@ADVPN-HUB# set file messages any any
root@ADVPN-HUB# set file messages authorization info
root@ADVPN-HUB# set file messages change-log none
root@ADVPN-HUB# set file messages interactive-commands none
root@ADVPN-HUB# set file messages structured-data
root@ADVPN-HUB# set file interactive-commands interactive-commands any
root@ADVPN-HUB# top
```

Commit and wait for the other spoke to be configured before you can verify the topology:

```
root@ADVPN-HUB# commit
```

Configure the Spokes

Configure all spokes, one at a time, using the same procedure for each device. The text in boldface is unique per your device. Refer to the lab topology in Figure 4.1.

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming ssh access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-SPOKE-1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces, here we set the MTU on the ST interface to 1400 to get fragmentation issues on the VPN:

```
root@host# set interfaces ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set interfaces ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set interfaces ge-0/0/4 unit 0 description voip family inet address 21.10.101.1/24
root@host# set interfaces st0 unit 0 description advpn multipoint family inet address
192.168.100.101/24
root@host# set interfaces st0 unit 0 family inet mtu 1400
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure the security zone UNTRUST, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
```

Configure the security zone CLIENTS, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# top edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# top edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# top edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# top
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the changes:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-SPOKE-1# set routing-options router-id 192.168.100.101
```

Configure the OSPF instance:

```
root@ADVPN-SPOKE-1# edit protocols ospf area 0.0.0.0
root@ADVPN-SPOKE-1# set interface st0.0 interface-type p2mp
root@ADVPN-SPOKE-1# set interface st0.0 metric 10
root@ADVPN-SPOKE-1# set interface st0.0 demand-circuit
root@ADVPN-SPOKE-1# set interface st0.0 flood-reduction
root@ADVPN-SPOKE-1# set interface st0.0 dynamic-neighbors
root@ADVPN-SPOKE-1# set interface st0.0 retransmit-interval 1
root@ADVPN-SPOKE-1# set interface st0.0 dead-interval 40
root@ADVPN-SPOKE-1# set interface ge-0/0/3.0 passive
root@ADVPN-SPOKE-1# set interface ge-0/0/4.0 passive
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@ADVPN-SPOKE-1# top edit security pki ca-profile Our-CA_Server
root@ADVPN-SPOKE-1# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@ADVPN-SPOKE-1# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@ADVPN-SPOKE-1# set revocation-check use-ocsp ocsp url http://1.1.1.2/ocsp
root@ADVPN-SPOKE-1# set revocation-check use-ocsp ocsp nonce-payload enable
```

Keep in mind that the `challenge-password` is unique to your CA:

```
root@ADVPN- SPOKE-1# top edit security pki auto-re-enrollment certificate-id ADVPN-PEER
root@ADVPN- SPOKE-1# set ca-profile-name Our-CA_Server
root@ADVPN- SPOKE-1# set re-generate-keypair
root@ADVPN- SPOKE-1# set re-enroll-trigger-time-percentage 10
root@ADVPN- SPOKE-1# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN- SPOKE-1# set scep-encryption-algorithm des3
root@ADVPN- SPOKE-1# top
```

Activate this configuration and then verify IP connectivity before continuing:

```
root@ADVPN-SPOKE-1# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@ADVPN-SPOKE-1# run ping 2.2.1.254
root@ADVPN-SPOKE-1# run ping 2.2.2.254
root@ADVPN-SPOKE-1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@ADVPN-SPOKE-1# set security address-book global address VOIP-Networks 21.20.0.0/16
root@ADVPN-SPOKE-1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Now define a policy for each direction between VOIP and ADVPN:

```
root@ADVPN-SPOKE-1# edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# top edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
```

```
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Define a policy for each direction between CLIENTS and ADVPN:

```
root@ADVPN-SPOKE-1# top edit security policies from-zone CLIENTS to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# top edit security policies from-zone ADVPN to-zone CLIENTS policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# top edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address any
root@ADVPN-SPOKE-1# set match destination-address any
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@ADVPN-SPOKE-1# set security policies default-policy deny-all

root@ADVPN-SPOKE-1# edit security policies global
root@ADVPN-SPOKE-1# set policy 1 match source-address any
root@ADVPN-SPOKE-1# set policy 1 match destination-address any
root@ADVPN-SPOKE-1# set policy 1 match application any
root@ADVPN-SPOKE-1# set policy 1 then deny
root@ADVPN-SPOKE-1# set policy 1 then log session-init session-close
root@ADVPN-SPOKE-1# top
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@ADVPN-SPOKE-1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate.

```
root@ADVPN-SPOKE-1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@ADVPN-SPOKE-1# run request security pki generate-key-pair certificate-id ADVPN-PEER size 2048 type
rsa
```

When you enroll your local-certificate you need to add some input (this was explained earlier at this same stage with the the hub site):

```
root@ADVPN-SPOKE-1# run request security pki local-certificate enroll ca-profile Our-CA_Server
certificate-id ADVPN-PEER domain-name advpn-spoke-1.mydomain.com ip-address 2.2.1.1 subject DC=mydomain.
com,L=Stockholm,O=Mydomain,OU=LAB,CN=advpn-spokes-1 challenge-password
8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-SPOKE-1# run request security pki local-certificate verify certificate-id ADVPN-PEER
CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status

root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, that will be explained in the Troubleshooting section later in this chapter.

The proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-SPOKE-1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set authentication-method rsa-signatures
root@ADVPN-SPOKE-1# set dh-group group14
root@ADVPN-SPOKE-1# set authentication-algorithm sha-256
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 28800
```

IKE Policy

In this example we use `ADVPN` as the name. If you had or were planning more AD-VPNs, then plan the name accordingly, for example, `ADVPN-Prim` and `ADVPN-Sec`:

```
root@ADVPN-SPOKE-1# top edit security ike policy ADVPN
root@ADVPN-SPOKE-1# set mode main
root@ADVPN-SPOKE-1# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set certificate local-certificate ADVPN-SPOKE-1
root@ADVPN-SPOKE-1# set certificate peer-certificate-type x509-signature
```

IKE Gateway

```
root@ADVPN-SPOKE-1# top edit security ike gateway ADVPN-HUB
root@ADVPN-SPOKE-1# set ike-policy ADVPN
root@ADVPN-SPOKE-1# set address 1.1.1.1
root@ADVPN-SPOKE-1# set dead-peer-detection probe-idle-tunnel
root@ADVPN-SPOKE-1# set local-identity distinguished-name
root@ADVPN-SPOKE-1# set remote-identity distinguished-name container CN=advpn-hub
root@ADVPN-SPOKE-1# set external-interface ge-0/0/1.0
root@ADVPN-SPOKE-1# set advpn suggester disable
root@ADVPN-SPOKE-1# set advpn partner connection-limit 10
root@ADVPN-SPOKE-1# set advpn partner idle-time 60
root@ADVPN-SPOKE-1# set advpn partner idle-threshold 5
root@ADVPN-SPOKE-1# set version v2-only
```

IPsec Proposal

```
root@ADVPN-SPOKE-1# top edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-SPOKE-1# set protocol esp
root@ADVPN-SPOKE-1# set authentication-algorithm hmac-sha-256-128
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 3600
```

IPsec Policy

```
root@ADVPN-SPOKE-1# top edit security ipsec policy ADVPN
root@ADVPN-SPOKE-1# set perfect-forward-secrecy keys group14
root@ADVPN-SPOKE-1# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

```
root@ADVPN-SPOKE-1# top edit security ipsec vpn ADVPN-HUB
root@ADVPN-SPOKE-1# set bind-interface st0.0
root@ADVPN-SPOKE-1# set ike gateway ADVPN-HUB
root@ADVPN-SPOKE-1# set ike ipsec-policy ADVPN
root@ADVPN-SPOKE-1# set establish-tunnels immediately
root@ADVPN-SPOKE-1# top
```

And add syslog for troubleshooting:

```
root@ADVPN-SPOKE-1# set system syslog user * any emergency
root@ADVPN-SPOKE-1# edit system syslog
root@ADVPN-SPOKE-1# set file messages any any
root@ADVPN-SPOKE-1# set file messages authorization info
root@ADVPN-SPOKE-1# set file messages change-log none
root@ADVPN-SPOKE-1# set file messages interactive-commands none
root@ADVPN-SPOKE-1# set file messages structured-data
root@ADVPN-SPOKE-1# set file interactive-commands interactive-commands any
root@ADVPN-SPOKE-1# top
```

Commit the configuration:

```
root@ADVPN-SPOKE-1# commit
```

Verification

Verify that your configuration is working. If something is not working, go to the troubleshooting section under ADVPN. Each local site should have a client from which you send traffic if you want to verify that traffic is floating through the system, otherwise you need to configure local policies to allow the Junos host to send traffic between certain zones.

On HUB:

This should give you SAs for all the spokes. Index and cookies will change:

```
root@ADVPN-HUB# run show security ike security-associations
Index      State  Initiator cookie    Responder cookie    Mode Remote Address
6139898    UP  187fa00207366fa3 cf455c90efc250da   IKEv2  2.2.2.1
6139899    UP  b760f86fb8b5b1ac c927570d355e6945 IKEv2  2.2.1.1
```

Verify that you see all your OSPF neighbors and have full state on all of them:

```
root@ADVPN-HUB# run show ospf neighbor
Address      Interface  State  ID     Pri Dead
192.168.100.101  st0.0    Full  192.168.100.101  128 -
192.168.100.102  st0.0    Full  192.168.100.102  128 -
```

When you see that you have neighborships with all peers, you should verify that you received all routes:

```
root@ADVPN-HUB# run show route protocol ospf
inet.0: 17 destinations, 18 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.101.0/24 *[OSPF/10] 00:25:13, metric 11
               > to 192.168.100.101 via st0.0
11.10.102.0/24 *[OSPF/10] 00:46:44, metric 11
               > to 192.168.100.102 via st0.0
21.10.101.0/24 *[OSPF/10] 00:25:13, metric 11
               > to 192.168.100.101 via st0.0
21.10.102.0/24 *[OSPF/10] 00:46:44, metric 11
               > to 192.168.100.102 via st0.0
192.168.100.101/32 *[OSPF/10] 00:25:13, metric 10
               > to 192.168.100.101 via st0.0
192.168.100.102/32 *[OSPF/10] 00:46:44, metric 10
               > to 192.168.100.102 via st0.0
224.0.0.5/32   *[OSPF/10] 01:12:09, metric 1
               MultiRecv
```

On Spokes

This should give you the HUB SAs. Index and cookies will change:

```
root@ADVPN-SPOKE-1# run show security ike security-associations
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275808    UP   b760f86fb8b5b1ac c927570d355e6945 IKEv2  1.1.1.1
```

When you see that you have neighborship with the HUB, you should verify that you received all routes and that they are pointing to the HUB:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24     *[OSPF/10] 00:42:24, metric 11
                > to 192.168.100.254 via st0.0
11.10.102.0/24   *[OSPF/10] 00:03:23, metric 21
                > to 192.168.100.254 via st0.0
21.10.0.0/24     *[OSPF/10] 00:42:24, metric 11
                > to 192.168.100.254 via st0.0
21.10.102.0/24   *[OSPF/10] 00:03:23, metric 21
                > to 192.168.100.254 via st0.0
192.168.100.102/32 *[OSPF/10] 00:03:23, metric 20
                > to 192.168.100.254 via st0.0
192.168.100.254/32 *[OSPF/10] 00:42:24, metric 10
                > to 192.168.100.254 via st0.0
224.0.0.5/32     *[OSPF/10] 00:54:35, metric 1
                MultiRecv
```

When you have traffic between the spokes it will give you the shortcut SAs. Index and cookies will change. You will see the remote spoke's address under `Remote Address`:

```
root@ADVPN-SPOKE-1# run show security ike security-associations sa-type shortcut
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275809    UP   360778f922e91e82 50f18f3d9163d410 IKEv2  2.2.2.1
```

When you have an active shortcut tunnel, you can see that you now have another path to the destination as compared to before the shortcut tunnel. And that's the path to the other spoke:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24       *[OSPF/10] 00:49:23, metric 11
                  > to 192.168.100.254 via st0.0
11.10.102.0/24 *[OSPF/10] 00:00:05, metric 11
                  > to 192.168.100.102 via st0.0
21.10.0.0/24       *[OSPF/10] 00:49:23, metric 11
                  > to 192.168.100.254 via st0.0
21.10.102.0/24 *[OSPF/10] 00:00:05, metric 11
                  > to 192.168.100.102 via st0.0
192.168.100.102/32 *[OSPF/10] 00:00:05, metric 10
                  > to 192.168.100.102 via st0.0
192.168.100.254/32 *[OSPF/10] 00:49:23, metric 10
                  > to 192.168.100.254 via st0.0
224.0.0.5/32       *[OSPF/10] 01:01:34, metric 1
                  MultiRecv
```

ADVPN with Redundant Internet Connection at Hub

The ADVPN use case has two remote offices with IPsec tunnels to headquarters but also has shortcut tunnels between the remote offices. Each site has two segments. At the headquarters there is a redundant Internet connection, but no BGP peering between the ISPs. We'll use a basic failover only if the primary connection loses its connection.

In Figure 4.2 you can only see two spokes, but you can add more in the same way as the two shown here. Keep in mind that you might need to change the `connection-limit` under each partner if you increase the network to a large scale.

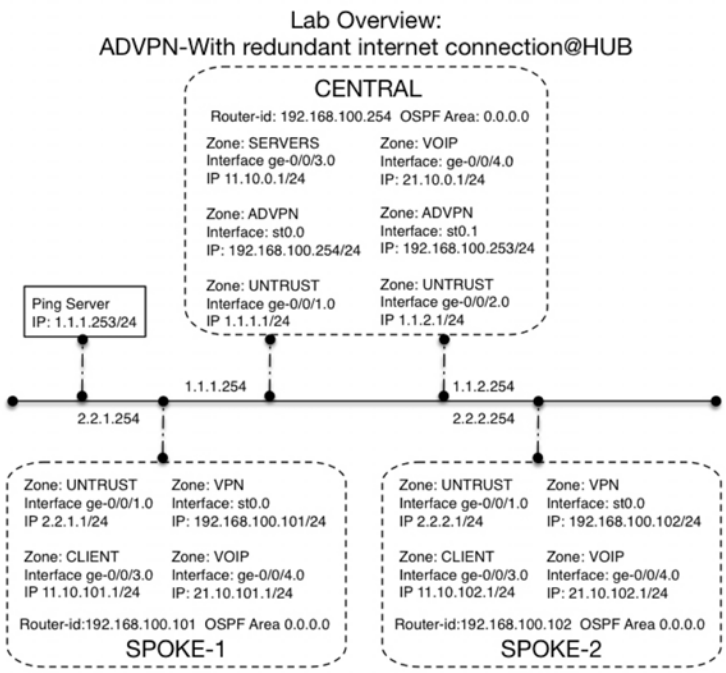


Figure 4.2 Auto Discovery VPN with Redundant Internet Connection at Hub

Requirements

Hardware: Juniper SRX Service Gateway SRX1xx – 650 for Spoke and Suggester deployment, SRX1K to SRX5K can only act as Suggester.

Software: Junos 12.3X48D10 and above

Routing: OSPF in the VPN topology

Certificate Authority: We will use SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, be sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-HUB
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces, and set the MTU on the ST interface to 1400 because we want to get fragmentation issues on the VPN:

```
root@host# edit interfaces
root@host# set ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
```

```
root@host# set ge-0/0/2 unit 0 description untrust family inet address 1.1.2.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set ge-0/0/4 unit 0 description voip family inet address 21.10.0.1/24
root@host# set st0 unit 0 description advpn multipoint family inet address 192.168.100.254/24
root@host# set st0 unit 1 description advpn multipoint family inet address 192.168.100.253/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# set st0 unit 1 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# edit routing-options static
root@host# set route 0.0.0.0/0 qualified-next-hop 1.1.1.254 metric 10
root@host# set route 0.0.0.0/0 qualified-next-hop 1.1.2.254 metric 11
```

Because there are two ISPs and we can't have two active default gateways, you need a way to monitor the primary connection. If the primary connection fails, you need to failover the default gateway to the secondary IPS. This is done with `IP-route` monitoring. Be sure that the target address is a host that is normally always up and responds to pings. Use the ping server located on the 1.1.1.0/24 network:

```
root@host# edit services rpm probe ISP1 test IP-ROUTE
root@hosy# set target address 1.1.1.253
root@host# set probe-count 5
root@host# set probe-interval 2
root@host# set test-interval 30
root@host# set thresholds successive-loss 5
root@host# set thresholds total-loss 5
root@host# set destination-interface ge-0/0/1.0
root@host# edit services ip-monitoring policy IP-ROUTE
root@host# set match rpm-probe ISP1
root@host# set then preferred-route route 0.0.0.0/0 next-hop 1.1.2.254
root@host# set then preferred-route route 0.0.0.0/0 preferred-metric 5
```

Configure the security zone UNTRUST, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ping
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.1 host-inbound-traffic protocols ospf
root@host# set interfaces st0.1 host-inbound-traffic system-services ping
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the configuration:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-HUB# set routing-options router-id 192.168.100.254
```

Configure OSPF:

```
root@ADVPN-HUB# edit protocols ospf area 0.0.0.0
root@ADVPN-HUB# set interface st0.0 interface-type p2mp
root@ADVPN-HUB# set interface st0.0 metric 10
root@ADVPN-HUB# set interface st0.0 demand-circuit
root@ADVPN-HUB# set interface st0.0 flood-reduction
root@ADVPN-HUB# set interface st0.0 dynamic-neighbors
root@ADVPN-HUB# set interface st0.0 retransmit-interval 1
root@ADVPN-HUB# set interface st0.0 dead-interval 40
root@ADVPN-HUB# set interface st0.1 interface-type p2mp
root@ADVPN-HUB# set interface st0.1 metric 10
root@ADVPN-HUB# set interface st0.1 demand-circuit
root@ADVPN-HUB# set interface st0.1 flood-reduction
root@ADVPN-HUB# set interface st0.1 dynamic-neighbors
root@ADVPN-HUB# set interface st0.1 retransmit-interval 1
root@ADVPN-HUB# set interface st0.1 dead-interval 40
root@ADVPN-HUB# set interface ge-0/0/3.0 passive
root@ADVPN-HUB# set interface ge-0/0/4.0 passive
```

Configure and request the certificates that you need to establish the tunnel. First you need to define how to find the CA:

```
root@ADVPN-HUB# edit security pki ca-profile Our-CA_Server
root@ADVPN-HUB# set ca-identity MydomainCertificateAuthority
```

Specify the CA SCEP URL:

```
root@ADVPN-HUB# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@ADVPN-HUB# set revocation-check use-ocsp ocsp nonce-payload enable
root@ADVPN-HUB# set revocation-check ocsp url http://11.10.0.10/ocsp
```

Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-HUB# edit security pki auto-re-enrollment certificate-id ADVPN-HUB
root@ADVPN-HUB# set ca-profile-name Our-CA_Server
root@ADVPN-HUB# set re-generate-keypair
root@ADVPN-HUB# set re-enroll-trigger-time-percentage 10
root@ADVPN-HUB# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN-HUB# set scep-encryption-algorithm des3
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@ADVPN-HUB# commit
```

Verify the gateway using an ICMP ping:

```
root@ADVPN-HUB# run ping 1.1.1.254
root@ADVPN-HUB# run ping 2.2.1.254
root@ADVPN-HUB# run ping 2.2.2.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the Destination NAT request:

```
root@ADVPN-HUB# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Configure the Destination NAT rule configuration:

```
root@ADVPN-HUB# edit security nat destination
root@ADVPN-HUB# set pool CertificateAuthority address 11.10.0.10/32
root@ADVPN-HUB# edit rule-set CertificateAuthority

root@ADVPN-HUB# set from interface ge-0/0/1.0
root@ADVPN-HUB# set rule 1 match source-address 0.0.0.0/0
root@ADVPN-HUB# set rule 1 match destination-address 1.1.1.2/32
root@ADVPN-HUB# set rule 1 then destination-nat pool CertificateAuthority
```

Configure the different address book objects that you will use later on to build your security policies:

```
root@ADVPN-HUB# edit security address-book global
root@ADVPN-HUB# set address VOIP-Networks 21.20.0.0/16
root@ADVPN-HUB# set address DATA-Networks 11.10.0.0/16
root@ADVPN-HUB# set address CertificateAuthority 11.10.0.10/32
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the `session-init` statement for logging to reduce log size.

First, build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@ADVPN-HUB# edit security policies from-zone UNTRUST to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address CertificateAuthority
root@ADVPN-HUB# set match application junos-http
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Define a policy for each direction between CLIENTS and ADVPN:

```
root@ADVPN-HUB# edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Now define a policy for each direction between SERVERS and VPN:

```
root@ADVPN-HUB# edit security policies from-zone SERVERS to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address any
```

```
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

You need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@ADVPN-HUB# set security policies default-policy deny-all

root@ADVPN-HUB# edit security policies global
root@ADVPN-HUB# set policy 1 match source-address any
root@ADVPN-HUB# set policy 1 match destination-address any
root@ADVPN-HUB# set policy 1 match application any
root@ADVPN-HUB# set policy 1 then deny
root@ADVPN-HUB# set policy 1 then log session-init session-close
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@ADVPN-HUB# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@ADVPN-HUB# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`:

```
root@ADVPN-HUB# run request security pki generate-key-pair certificate-id ADVPN-HUB size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-HUB# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
ADVPN-HUB domain-name advpn-hub.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Stockholm,O
=Mydomain,OU=LAB,CN=advpn-hub challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to: http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-HUB# run request security pki local-certificate verify certificate-id ADVPN-HUB

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-HUB# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set authentication-method rsa-signatures
root@ADVPN-HUB# set dh-group group14
```

```
root@ADVPN-HUB# set authentication-algorithm sha-256
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 28800
```

IKE Policy

The name indicates that it's used for ADVPN topologies, and this device should only serve as a Suggester (hub). Lets bind the local-certificate to this policy:

```
root@ADVPN-HUB# edit security ike policy ADVPN
root@ADVPN-HUB# set mode main
root@ADVPN-HUB# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set certificate local-certificate ADVPN-HUB
root@ADVPN-HUB# set certificate peer-certificate-type x509-signature
```

IKE Gateway

Configure how to authenticate the spokes that you want to establish an ADVPN tunnel to. Here, named `ADVPN-SPOKES` for the remote peers. The `wildcard OU` name, is what to search for in the remote peer's certificate to authenticate the peer. The IKE gateway will be bound together in the IPsec VPN configuration:

```
root@ADVPN-HUB# edit security ike gateway ADVPN-SPOKES-PRI
root@ADVPN-HUB# set ike-policy ADVPN
root@ADVPN-HUB# set dynamic distinguished-name wildcard OU=LAB
root@ADVPN-HUB# set dynamic ike-user-type group-ike-id
root@ADVPN-HUB# set local-identity distinguished-name
root@ADVPN-HUB# set external-interface ge-0/0/1.0
root@ADVPN-HUB# set advpn partner disable
root@ADVPN-HUB# set version v2-only

root@ADVPN-HUB# edit security ike gateway ADVPN-SPOKES-SEC
root@ADVPN-HUB# set ike-policy ADVPN
root@ADVPN-HUB# set dynamic distinguished-name wildcard OU=LAB
root@ADVPN-HUB# set dynamic ike-user-type group-ike-id
root@ADVPN-HUB# set local-identity distinguished-name
root@ADVPN-HUB# set external-interface ge-0/0/2.0
root@ADVPN-HUB# set advpn partner disable
root@ADVPN-HUB# set version v2-only
```

IPsec Proposal

Here are the parameters used in the IPsec policies:

```
root@ADVPN-HUB# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-HUB# set protocol esp
root@ADVPN-HUB# set authentication-algorithm hmac-sha-256-128
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 3600
```

IPsec Policy

The IPsec policy binds IPsec proposals to be used in the IPsec VPN configuration:

```
root@ADVPN-HUB# edit security ipsec policy ADVPN
root@ADVPN-HUB# set perfect-forward-secrecy keys group14
root@ADVPN-HUB# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

Now bind together the IKE Gateway and the IPsec Policy with the secure tunnel interface. They are called `ADVPN-SPOKES-PRI` and `ADVPN-SPOKES-SEC` to match the IKE gateway names:

```
root@ADVPN-HUB# edit security ipsec vpn ADVPN-SPOKES-PRI
root@ADVPN-HUB# set bind-interface st0.0
root@ADVPN-HUB# set ike gateway ADVPN-SPOKES-PRI
root@ADVPN-HUB# set ike ipsec-policy ADVPN

root@ADVPN-HUB# edit security ipsec vpn ADVPN-SPOKES-SEC
root@ADVPN-HUB# set bind-interface st0.1
root@ADVPN-HUB# set ike gateway ADVPN-SPOKES-SEC
root@ADVPN-HUB# set ike ipsec-policy ADVPN
```

Add syslog for troubleshooting:

```
root@ADVPN-HUB# set system syslog user * any emergency
root@ADVPN-HUB# edit system syslog
root@ADVPN-HUB# set file messages any any
root@ADVPN-HUB# set file messages authorization info
root@ADVPN-HUB# set file messages change-log none
root@ADVPN-HUB# set file messages interactive-commands none
root@ADVPN-HUB# set file messages structured-data
root@ADVPN-HUB# set file interactive-commands interactive-commands any
```

Commit and wait for the other spoke to be configured before you can verify the topology:

```
root@ADVPN-HUB# commit
```

Configure the Spokes

Configure all spokes, one at a time, using the same procedure for each device. Text in boldface is unique per device. Refer to the lab topology in Figure 4.1.

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-SPOKE-1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

root@host# set system root-authentication plain-text-password

```
New password:
Retype new password:
```

Configure all network interfaces, here we set the MTU on the ST interface to 1400 to get fragmentation issues on the VPN:

```
root@host# edit interfaces
root@host# set ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set ge-0/0/4 unit 0 description voip family inet address 21.10.101.1/24
root@host# set st0 unit 0 description advpn multipoint family inet address 192.168.100.101/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# set routing-options static route 0.0.0.0/0 next-hop 2.2.1.254
```

Configure the security zone UNTRUST, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
```

Configure the security zone CLIENTS, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the changes:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-SPOKE-1# set routing-options router-id 192.168.100.101
```

Configure the OSPF instance:

```
root@ADVPN-SPOKE-1# edit protocols ospf area 0.0.0.0
root@ADVPN-SPOKE-1# set interface st0.0 interface-type p2mp
root@ADVPN-SPOKE-1# set interface st0.0 metric 10
root@ADVPN-SPOKE-1# set interface st0.0 demand-circuit
root@ADVPN-SPOKE-1# set interface st0.0 flood-reduction
root@ADVPN-SPOKE-1# set interface st0.0 dynamic-neighbors
root@ADVPN-SPOKE-1# set interface st0.0 retransmit-interval 1
root@ADVPN-SPOKE-1# set interface st0.0 dead-interval 40
root@ADVPN-SPOKE-1# set interface ge-0/0/3.0 passive
root@ADVPN-SPOKE-1# set interface ge-0/0/4.0 passive
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@ADVPN-SPOKE-1# edit security pki ca-profile Our-CA_Server
root@ADVPN-SPOKE-1# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@ADVPN-SPOKE-1# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Keep in mind that the `challenge-password` is unique to your CA:

```
root@ADVPN-SPOKE-1# set revocation-check ocsp url http://1.1.1.2/ocsp
root@ADVPN-SPOKE-1# set revocation-check use-ocsp ocsp nonce-payload enable
root@ADVPN-SPOKE-1# edit security pki auto-re-enrollment certificate-id ADVPN-HUB
root@ADVPN-SPOKE-1# set ca-profile-name Our-CA_Server
```

```
root@ADVPN-SPOKE-1# set re-generate-keypair
root@ADVPN-SPOKE-1# set re-enroll-trigger-time-percentage 10
root@ADVPN-SPOKE-1# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN-SPOKE-1# set scep-encryption-algorithm des3
root@ADVPN-SPOKE-1# top
```

Activate this configuration and then verify IP connectivity before continuing:

```
root@ADVPN-SPOKE-1# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@ADVPN-SPOKE-1# run ping 2.2.1.254
root@ADVPN-SPOKE-1# run ping 2.2.2.254
root@ADVPN-SPOKE-1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@ADVPN-SPOKE-1# set security address-book global address VOIP-Networks 21.20.0.0/16
root@ADVPN-SPOKE-1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Now define a policy for each direction between VOIP and ADVPN:

```
root@ADVPN-SPOKE-1# edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Define a policy for each direction between CLIENTS and ADVPN:

```
root@ADVPN-SPOKE-1# edit security policies from-zone CLIENTS to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone CLIENTS policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

```
root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address any
root@ADVPN-SPOKE-1# set match destination-address any
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@ADVPN-SPOKE-1# set security policies default-policy deny-all

root@ADVPN-SPOKE-1# edit security policies global
root@ADVPN-SPOKE-1# set policy 1 match source-address any
root@ADVPN-SPOKE-1# set policy 1 match destination-address any
root@ADVPN-SPOKE-1# set policy 1 match application any
root@ADVPN-SPOKE-1# set policy 1 then deny
root@ADVPN-SPOKE-1# set policy 1 then log session-init session-close
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@ADVPN-SPOKE-1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@ADVPN-HUB# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@ADVPN-SPOKE-1# run request security pki generate-key-pair certificate-id ADVPN-PEER size 2048 type
rsa
```

When you enroll your local-certificate you need to add some input:

```
root@ADVPN-SPOKE-1# run request security pki local-certificate enroll ca-profile Our-CA_Server
certificate-id ADVPN-PEER domain-name advpn-spoke-1.mydomain.com ip-address 2.2.1.1 subject
DC=mydomain.com,L=Stockholm,O=Mydomain,OU=LAB,CN=advpn-spokes-1 challenge-password
8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-SPOKE-1# run request security pki local-certificate verify certificate-id ADVPN-PEER

CA certificate Our-CA_Server: Revocation check is in progress. Please check the PKId debug logs for
completion status

root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-SPOKE-1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set authentication-method rsa-signatures
root@ADVPN-SPOKE-1# set dh-group group14
root@ADVPN-SPOKE-1# set authentication-algorithm sha-256
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 28800
```

IKE Policy

In this example we use `ADVPN` as the name. If you had or were planning more ADVPNs, then plan the name accordingly, for example, `ADVPN-Prim` and `ADVPN-Sec`:

```
root@ADVPN-SPOKE-1# edit security ike policy ADVPN
root@ADVPN-SPOKE-1# set mode main
root@ADVPN-SPOKE-1# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set certificate local-certificate ADVPN-SPOKE-1
root@ADVPN-SPOKE-1# set certificate peer-certificate-type x509-signature
```

IKE Gateway

```
root@ADVPN-SPOKE-1# edit security ike gateway ADVPN-HUB
root@ADVPN-SPOKE-1# set ike-policy ADVPN
root@ADVPN-SPOKE-1# set address 1.1.1.1
root@ADVPN-SPOKE-1# set address 1.1.2.1
root@ADVPN-SPOKE-1# set dead-peer-detection probe-idle-tunnel
root@ADVPN-SPOKE-1# set local-identity distinguished-name
root@ADVPN-SPOKE-1# set remote-identity distinguished-name container CN=advpn-hub
root@ADVPN-SPOKE-1# set external-interface ge-0/0/1.0
root@ADVPN-SPOKE-1# set advpn suggester disable
root@ADVPN-SPOKE-1# set advpn partner connection-limit 10
root@ADVPN-SPOKE-1# set advpn partner idle-time 60
root@ADVPN-SPOKE-1# set advpn partner idle-threshold 5
root@ADVPN-SPOKE-1# set version v2-only
```

IPsec Proposal

```
root@ADVPN-SPOKE-1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-SPOKE-1# set protocol esp
root@ADVPN-SPOKE-1# set authentication-algorithm hmac-sha-256-128
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 3600
```

IPsec Policy

```
root@ADVPN-SPOKE-1# edit security ipsec policy ADVPN
root@ADVPN-SPOKE-1# set perfect-forward-secrecy keys group14
root@ADVPN-SPOKE-1# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

```
root@ADVPN-SPOKE-1# edit security ipsec vpn ADVPN-HUB
root@ADVPN-SPOKE-1# set bind-interface st0.0
root@ADVPN-SPOKE-1# set ike gateway ADVPN-HUB
root@ADVPN-SPOKE-1# set ike ipsec-policy ADVPN
root@ADVPN-SPOKE-1# set establish-tunnels immediately
```

And add syslog for troubleshooting:

```
root@ADVPN-SPOKE-1# set system syslog user * any emergency
root@ADVPN-SPOKE-1# edit system syslog
root@ADVPN-SPOKE-1# set file messages any any
root@ADVPN-SPOKE-1# set file messages authorization info
root@ADVPN-SPOKE-1# set file messages change-log none
root@ADVPN-SPOKE-1# set file messages interactive-commands none
root@ADVPN-SPOKE-1# set file messages structured-data
root@ADVPN-SPOKE-1# set file interactive-commands interactive-commands any
```

Commit the configuration:

```
root@ADVPN-SPOKE-1# commit
```

Verification

Let's verify that the configuration is working. If something is not working, go to the troubleshooting section for ADVPN.

On the HUB

This should give you SAs for all the spokes. Index and cookies will change:

```
the root@ADVPN-HUB# run show security ike security-associations
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
6139907    UP 164749a9f9f6d00a 4d5f206101f4088c IKEv2  2.2.1.1
6139908    UP 28c5fc6d4dca14ab 0ce428bdfde37a0d IKEv2  2.2.2.1
```

Verify that we see all our OSPF neighbors and have full state on all of them. The address shows the next-hop and the ID shows the router ID from the `routing-options` stanza:

```
root@ADVPN-HUB# run show ospf neighbor
Address      Interface  State  ID   Pri Dead
192.168.100.102  st0.0  Full 192.168.100.102  128 -
192.168.100.101  st0.0  Full 192.168.100.101  128 -
```

When we see that you have OSPF neighborship with all peers, you should verify that you received all routes, and that each spoke has a different next-hop.

```
root@ADVPN-HUB# run show route protocol ospf
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

11.10.101.0/24   *[OSPF/10] 16:53:36, metric 11
> to 192.168.100.101 via st0.0
11.10.102.0/24   *[OSPF/10] 16:25:42, metric 11
> to 192.168.100.102 via st0.0
21.10.101.0/24   *[OSPF/10] 16:53:36, metric 11
> to 192.168.100.101 via st0.0
21.10.102.0/24   *[OSPF/10] 16:25:42, metric 11
> to 192.168.100.102 via st0.0
192.168.100.101/32 *[OSPF/10] 16:53:36, metric 10
> to 192.168.100.101 via st0.0
192.168.100.102/32 *[OSPF/10] 16:25:42, metric 10
> to 192.168.100.102 via st0.0
224.0.0.5/32   *[OSPF/10] 20:35:43, metric 1
MultiRecv
```

Verify active default route (the active route marked in bold due to lower metric). If you do not have your primary default gateway up, go to the troubleshooting section.

```
root@ADVPN-HUB# run show route 0.0.0.0
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0  *[Static/5] 17:24:19, metric 10
> to 1.1.1.254 via ge-0/0/1.0
 [Static/5] 18:42:51, metric 11
> to 1.1.2.254 via ge-0/0/2.0
```

On Spokes

This should give you the HUB SAs. Index and cookies will change:

```
root@ADVPN-SPOKE-1# run show security ike security-associations
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275816    UP 164749a9f9f6d00a 4d5f206101f4088c IKEv2  1.1.1.1
```

Verify that we have full OSPF neighborship with the Hub.

```
root@ADVPN-SPOKE-1# run show ospf neighbor
Address        Interface  State  ID    Pri Dead
192.168.100.254    st0.0    Full 192.168.100.254    128 -
```

When you see that you have OSPF neighborship with the Hub you should verify that you received all routes and that they point to the Hub:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24   *[OSPF/10] 16:55:00, metric 11
> to 192.168.100.254 via st0.0
11.10.102.0/24   *[OSPF/10] 16:27:06, metric 21
> to 192.168.100.254 via st0.0
21.10.0.0/24   *[OSPF/10] 16:55:00, metric 11
> to 192.168.100.254 via st0.0
21.10.102.0/24   *[OSPF/10] 16:27:06, metric 21
> to 192.168.100.254 via st0.0
192.168.100.102/32 *[OSPF/10] 16:27:06, metric 20
> to 192.168.100.254 via st0.0
192.168.100.253/32 *[OSPF/10] 16:55:00, metric 10
> to 192.168.100.254 via st0.0
192.168.100.254/32 *[OSPF/10] 16:55:00, metric 10
> to 192.168.100.254 via st0.0
224.0.0.5/32   *[OSPF/10] 19:56:12, metric 1
MultiRecv
```

When you have traffic between the spokes it will give you the shortcut SAs. Index and cookies will change. You will see the remote spoke's address under `Remote Address`:

```
root@ADVPN-SPOKE-1# run show security ike security-associations sa-type shortcut
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275817    UP c47ecb3ac420b588 e0f48944fb66b456 IKEv2  2.2.2.1
```

When you have an active shortcut tunnel, you can see that you now have another path to the destination as compared to before the shortcut tunnel. You see that the next-hop has changed:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
11.10.0.0/24   *[OSPF/10] 16:55:48, metric 11
> to 192.168.100.254 via st0.0
11.10.102.0/24   *[OSPF/10] 00:00:05, metric 11
> to 192.168.100.102 via st0.0
```

```
21.10.0.0/24   *[OSPF/10] 16:55:48, metric 11
> to 192.168.100.254 via st0.0
21.10.102.0/24   *[OSPF/10] 00:00:05, metric 11
> to 192.168.100.102 via st0.0
192.168.100.102/32 *[OSPF/10] 00:00:05, metric 10
> to 192.168.100.102 via st0.0
192.168.100.253/32 *[OSPF/10] 16:55:48, metric 10
> to 192.168.100.254 via st0.0
192.168.100.254/32 *[OSPF/10] 16:55:48, metric 10
> to 192.168.100.254 via st0.0
224.0.0.5/32   *[OSPF/10] 19:57:00, metric 1
MultiRecv
```

ADVPN with Redundant Internet Connections at Hub-and-Spokes

This use case has two remote offices that need to build an IPsec tunnel to headquarters, but also allow shortcut tunnels between the remote offices. Each site has two segments. There are redundant Internet connections on both the headquarters and in the remote office, but none of them have BGP peering between the ISPs. We'll use a basic failover only if the primary connection loses its connection.

In Figure 4.3 you can only see two spokes but you can add more in the same way as described for the two spokes. Keep in mind that you might need to change the `connection-limit` under each partner if you increase the network to a large extent.

NOTE The configuration is displayed in snippets below. The configuration files are available on this book's landing page at http://www.juniper.net/dayone. Keep in mind that you need to change variable data such as interface, IP addresses, and other information related to your own network.

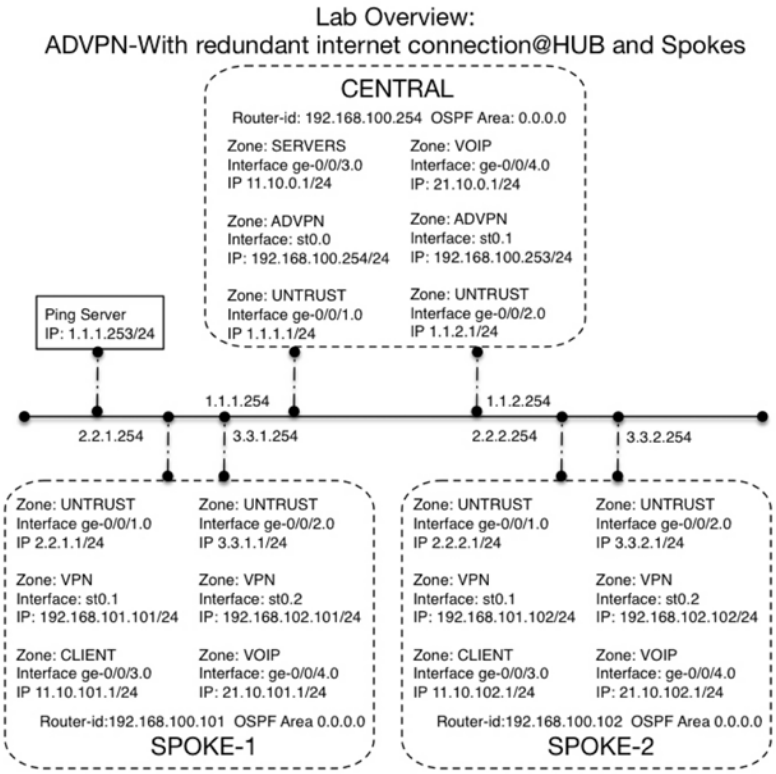


Figure 4.3 Auto Discovery VPN with Redundant Internet Connections at Hub-and-Spokes

Requirements

Hardware: Juniper SRX Service Gateway SRX1xx – 650 for Spoke and Suggester deployment, SRX1K to SRX5K can only act as Suggester.

Software: Junos 12.3X48D10 and above

Routing: OSPF in the VPN topology

Certificate Authority: We will use SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) for signing and revocation verification. It is also possible to use manual signing and certificate revocation lists, but those are not described in this book. Keep in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, be sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

CAUTION These step-by-step instructions will erase all current configurations!

Configure the Hub Site

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-HUB
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Because you erased all configurations, you must now set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces, and set the MTU on the ST interface to 1400 because we want to get fragmentation issues on the VPN:

```
root@host# edit interface
root@host# set ge-0/0/1 unit 0 description untrust family inet address 1.1.1.1/24
root@host# set ge-0/0/2 unit 0 description untrust family inet address 1.1.2.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.0.1/24
root@host# set ge-0/0/4 unit 0 description voip family inet address 21.10.0.1/24
root@host# set st0 unit 0 description advpn multipoint family inet address 192.168.100.254/24
root@host# set st0 unit 1 description advpn multipoint family inet address 192.168.100.253/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# set st0 unit 1 family inet mtu 1400
root@host# top
```

Configure a default route:

```
root@host# edit routing-options static
root@host# set route 0.0.0.0/0 qualified-next-hop 1.1.1.254 metric 10
root@host# set route 0.0.0.0/0 qualified-next-hop 1.1.2.254 metric 11
```

Because there are two ISPs and we can't have two active default gateways, we need a way to monitor the primary connection. If the primary connection fails, we need to failover our default gateway to the secondary IPS. This is done with `IP-route` monitoring. Be sure that the target address is a host that is normally always up and responds to pings. Use the ping server located on the 1.1.1.0/24 network:

```
root@host# edit services rpm probe ISP1 test IP-ROUTE
root@host# set target address 1.1.1.253
root@host# set probe-count 5
root@host# set probe-interval 2
root@host# set test-interval 30
root@host# set thresholds successive-loss 5
root@host# set thresholds total-loss 5
root@host# set destination-interface ge-0/0/1.0

root@host# edit services ip-monitoring policy IP-ROUTE
root@host# set match rpm-probe ISP1
root@host# set then preferred-route route 0.0.0.0/0 next-hop 1.1.2.254
root@host# set then preferred-route route 0.0.0.0/0 preferred-metric 5
```

Configure the security zone UNTRUST-1, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST-1
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
```

Configure the security zone UNTRUST-2, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST-2
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ping
```

Configure the security zone SERVERS, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone SERVERS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus any allowed incoming administrative services:

```
root@host# edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.1 host-inbound-traffic protocols ospf
root@host# set interfaces st0.1 host-inbound-traffic system-services ping
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the configuration:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-HUB# set routing-options router-id 192.168.100.254
```

Configure OSPF:

```
root@ADVPN-HUB# edit protocols ospf area 0.0.0.0
root@ADVPN-HUB# set interface st0.0 interface-type p2mp
root@ADVPN-HUB# set interface st0.0 metric 10
root@ADVPN-HUB# set interface st0.0 demand-circuit
root@ADVPN-HUB# set interface st0.0 flood-reduction
root@ADVPN-HUB# set interface st0.0 dynamic-neighbors
root@ADVPN-HUB# set interface st0.0 retransmit-interval 1
root@ADVPN-HUB# set interface st0.0 dead-interval 40
root@ADVPN-HUB# set interface st0.1 interface-type p2mp
root@ADVPN-HUB# set interface st0.1 metric 10
root@ADVPN-HUB# set interface st0.1 demand-circuit
root@ADVPN-HUB# set interface st0.1 flood-reduction
root@ADVPN-HUB# set interface st0.1 dynamic-neighbors
root@ADVPN-HUB# set interface st0.1 retransmit-interval 1
root@ADVPN-HUB# set interface st0.1 dead-interval 40
root@ADVPN-HUB# set interface ge-0/0/3.0 passive
root@ADVPN-HUB# set interface ge-0/0/4.0 passive
```

Configure and request the certificates that you need to establish the tunnel. First you need to define how to find the CA:

```
root@ADVPN-HUB# edit security pki ca-profile Our-CA_Server
root@ADVPN-HUB# set ca-identity MydomainCertificateAuthority
```

Specify the CA SCEP URL:

```
root@ADVPN-HUB# set enrollment url http://11.10.0.10/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@ADVPN-HUB# set revocation-check use-ocsp ocsp
root@ADVPN-HUB# set revocation-check ocsp url http://11.10.0.10/ocsp
```

Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-HUB# edit security pki auto-re-enrollment certificate-id ADVPN-HUB
root@ADVPN-HUB# set ca-profile-name Our-CA_Server
root@ADVPN-HUB# set re-generate-keypair
root@ADVPN-HUB# set re-enroll-trigger-time-percentage 10
root@ADVPN-HUB# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN-HUB# set scep-encryption-algorithm des3
root@ADVPN-HUB# top
```

Commit the configuration to verify your IP connectivity before continuing:

```
root@ADVPN-HUB# commit
```

Verify the gateway using an ICMP ping:

```
root@ADVPN-HUB# run ping 1.1.1.254
root@ADVPN-HUB# run ping 2.2.1.254
root@ADVPN-HUB# run ping 2.2.2.254
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

As the remote spokes need to access the CA for certificate enrollment, you also need to configure a firewall and NAT policy allowing external traffic to reach the CA.

First you need to configure the external interface to respond to ARP so the machine responds to the destination NAT request:

```
root@ADVPN-HUB# set interfaces ge-0/0/1 unit 0 description untrust family inet address 1.1.1.2/24
```

Configure the Destination NAT rule configuration:

```
root@ADVPN-HUB# edit security nat destination
root@ADVPN-HUB# set pool CertificateAuthority address 11.10.0.10/32
root@ADVPN-HUB# edit rule-set CertificateAuthority

root@ADVPN-HUB# set from interface ge-0/0/1.0
root@ADVPN-HUB# set rule 1 match source-address 0.0.0.0/0
root@ADVPN-HUB# set rule 1 match destination-address 1.1.1.2/32
root@ADVPN-HUB# set rule 1 then destination-nat pool CertificateAuthority
```

Configure the different address book objects that you will use later on to build your security policies:

```
root@ADVPN-HUB# edit security address-book global
root@ADVPN-HUB# set address VOIP-Networks 21.20.0.0/16
root@ADVPN-HUB# set address DATA-Networks 11.10.0.0/16
root@ADVPN-HUB# set address CertificateAuthority 11.10.0.10/32
```

Here you start building the security policy that will define what traffic should be allowed in different directions.

NOTE In a production environment, you can remove the `session-init` statement for logging to reduce log size.

First, build the policy that allows incoming traffic to the CA server from the remote spokes:

```
root@ADVPN-HUB# edit security policies from-zone UNTRUST-1 to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address CertificateAuthority
root@ADVPN-HUB# set match application junos-http
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Now define a policy for each direction between VOIP and ADVPN:

```
root@ADVPN-HUB# edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-HUB# set match source-address VOIP-Networks
root@ADVPN-HUB# set match destination-address VOIP-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

Now define a policy for each direction between SERVERS and ADVPN:

```
root@ADVPN-HUB# edit security policies from-zone SERVERS to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close

root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone SERVERS policy 1
root@ADVPN-HUB# set match source-address DATA-Networks
root@ADVPN-HUB# set match destination-address DATA-Networks
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

```
root@ADVPN-HUB# edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-HUB# set match source-address any
root@ADVPN-HUB# set match destination-address any
root@ADVPN-HUB# set match application any
root@ADVPN-HUB# set then permit
root@ADVPN-HUB# set then log session-init session-close
```

You need to be able to trace traffic that does not hit a firewall rule, so add a global config with a `deny` policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy, but for security precautions, you should add this policy:

```
root@ADVPN-HUB# set security policies default-policy deny-all
root@ADVPN-HUB# edit security policies global
root@ADVPN-HUB# set policy 1 match source-address any
root@ADVPN-HUB# set policy 1 match destination-address any
root@ADVPN-HUB# set policy 1 match application any
root@ADVPN-HUB# set policy 1 then deny
root@ADVPN-HUB# set policy 1 then log session-init session-close
```

The next step is to enroll the root certificate from our trusted CA, if the fingerprint is okay, type `Yes`:

```
root@ADVPN-HUB# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Now verify that you trust the downloaded certificate. Run the following command, and you can see the current numbers of valid OSCP verifications:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now let's verify that we trust the downloaded certificate:

```
root@ADVPN-HUB# run request security pki ca-certificate verify ca-profile Our-CA_Server
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

Wait to see that you have a successful verification, and if the number is not updated then you have a problem with the certificate or the OSCP response service:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

It's time to generate a `key-pair` for the `local-certificate`, like this:

```
root@ADVPN-HUB# run request security pki generate-key-pair certificate-id ADVPN-HUB size 2048 type rsa
```

When you enroll your `local-certificate`, you need to give some it a unique input per device. Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-HUB# run request security pki local-certificate enroll ca-profile Our-CA_Server certificate-id
ADVPN-HUB domain-name advpn-hub.mydomain.net ip-address 1.1.1.1 subject DC=mydomain.net,L=Sunnyvale,O
=Mydomain,OU=LAB,CN=advpn-hub challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
```

Certificate-id = The name used for the above generated key-pair.

Domain-name = FQDN of the box, corresponding to the IP-Address of the box.

IP-Address = IP-Address of the box corresponding to the FQDN.

Subject = DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit name>,O=<Organization name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>

Challenge-password = Your challenge password to authenticate to the CA server for your SCEP request. Received by going to url http://11.10.0.10/CertSrv/mscep_admin. Optionally, you can add encryption and hash for your SCEP request.

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-HUB# run request security pki local-certificate verify certificate-id ADVPN-HUB

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status

root@ADVPN-HUB# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, it's time to configure the IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-HUB# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set authentication-method rsa-signatures
root@ADVPN-HUB# set dh-group group14
root@ADVPN-HUB# set authentication-algorithm sha-256
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 28800
```

IKE Policy

The policy is used for ADVPN topologies, and this device should only serve as a Suggester (hub). We will bind the local-certificate to this policy:

```
root@ADVPN-HUB# edit security ike policy ADVPN
root@ADVPN-HUB# set mode main
root@ADVPN-HUB# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB# set certificate local-certificate ADVPN-HUB
root@ADVPN-HUB# set certificate peer-certificate-type x509-signature
```

IKE Gateway

Configure how to authenticate the spokes that you want to establish an ADVPN tunnel to. Here, named `ADVPN-SPOKES-PRI` for the remote peers depending on which IPs are in use. The `wildcard OU` name is what to search for in the remote peer's certificate to authenticate the peer. The IKE gateway will be bound together in the IPsec VPN configuration:

```
root@ADVPN-HUB# edit security ike gateway ADVPN-SPOKES-PRI
root@ADVPN-HUB# set ike-policy ADVPN
root@ADVPN-HUB# set dynamic distinguished-name wildcard OU=LAB
root@ADVPN-HUB# set dynamic ike-user-type group-ike-id
root@ADVPN-HUB# set local-identity distinguished-name
root@ADVPN-HUB# set external-interface ge-0/0/1.0
root@ADVPN-HUB# set advpn partner disable
root@ADVPN-HUB# set version v2-only

root@ADVPN-HUB# edit security ike gateway ADVPN-SPOKES-SEC
root@ADVPN-HUB# set ike-policy ADVPN
root@ADVPN-HUB# set dynamic distinguished-name wildcard OU=LAB
root@ADVPN-HUB# set dynamic ike-user-type group-ike-id
root@ADVPN-HUB# set local-identity distinguished-name
root@ADVPN-HUB# set external-interface ge-0/0/2.0
root@ADVPN-HUB# set advpn partner disable
root@ADVPN-HUB# set version v2-only
```

IPsec Proposal

Here are the parameters used in the IPsec policies:

```
root@ADVPN-HUB# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-HUB# set protocol esp
root@ADVPN-HUB# set authentication-algorithm hmac-sha-256-128
root@ADVPN-HUB# set encryption-algorithm aes-256-cbc
root@ADVPN-HUB# set lifetime-seconds 3600
```

IPsec Policy

The IPsec policy binds IPsec proposals to be used in the IPsec VPN configuration:

```
root@ADVPN-HUB# edit security ipsec policy ADVPN
root@ADVPN-HUB# set perfect-forward-secrecy keys group14
root@ADVPN-HUB# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

Now bind together the IKE Gateway and the IPsec Policy with the secure tunnel interface. VPN names are the same as for the IKE gateways:

```
root@ADVPN-HUB# edit security ipsec vpn ADVPN-SPOKES-PRI
root@ADVPN-HUB# set bind-interface st0.0
root@ADVPN-HUB# set ike gateway ADVPN-SPOKES-PRI
root@ADVPN-HUB# set ike ipsec-policy ADVPN

root@ADVPN-HUB# edit security ipsec vpn ADVPN-SPOKES-SEC
root@ADVPN-HUB# set bind-interface st0.1
root@ADVPN-HUB# set ike gateway ADVPN-SPOKES-SEC
root@ADVPN-HUB# set ike ipsec-policy ADVPN
```

Add syslog for troubleshooting:

```
root@ADVPN-HUB# set system syslog user * any emergency
root@ADVPN-HUB# edit system syslog
root@ADVPN-HUB# set file messages any any
root@ADVPN-HUB# set file messages authorization info
root@ADVPN-HUB# set file messages change-log none
root@ADVPN-HUB# set file messages interactive-commands none
root@ADVPN-HUB# set file messages structured-data
root@ADVPN-HUB# set file interactive-commands interactive-commands any
```

Commit and wait for the other spoke to be configured before you can verify the topology:

```
root@ADVPN-HUB# commit
```

Configure the Spokes

Configure all spokes, one at a time, using the same procedure for each device. Text in boldface is unique per device. Refer to the lab topology in Figure 4.1.

First erase all configurations so you can start with a clean configuration:

```
root@host# delete
```

Allow incoming `ssh` access for management of the device:

```
root@host# set system services ssh
```

Configure a hostname for this machine:

```
root@host# set system host-name ADVPN-SPOKE-1
```

Set the system clock:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs the correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

We erased all configurations, so now you must set a new root administrator password:

```
root@host# set system root-authentication plain-text-password
New password:
Retype new password:
```

Configure all network interfaces, here we set the MTU on the ST interface to 1400 to get fragmentation issues on the VPN:

```
root@host# edit interfaces
root@host# set ge-0/0/1 unit 0 description untrust family inet address 2.2.1.1/24
root@host# set ge-0/0/2 unit 0 description untrust family inet address 3.3.1.1/24
root@host# set ge-0/0/3 unit 0 description data family inet address 11.10.101.1/24
root@host# set ge-0/0/4 unit 0 description voip family inet address 21.10.101.1/24
root@host# set st0 unit 0 description advpn multipoint family inet address 192.168.100.101/24
root@host# set st0 unit 1 description advpn multipoint family inet address 192.168.100.201/24
root@host# set st0 unit 0 family inet mtu 1400
root@host# set st0 unit 1 family inet mtu 1400
```

Configure a default route:

```
root@host# edit routing-options static
root@host# set route 0.0.0.0/0 qualified-next-hop 2.2.1.254 metric 10
root@host# set route 0.0.0.0/0 qualified-next-hop 3.3.1.254 metric 11
```

Because there are two ISPs and you can't have two active default gateways, you need a way to monitor the primary connection. When the primary connection fails, we need to failover our default gateway to the secondary IPS. This is done with `IP-Route` monitoring. (Be sure that the target address is a host that is normally always up and responds to ping. We used the ping server located on the 1.1.1.0/24 network):

```
root@host# edit services rpm probe ISP1 test IP-ROUTE
root@host# set target address 1.1.1.253
root@host# set probe-count 5
root@host# set probe-interval 2
root@host# set test-interval 30
root@host# set thresholds successive-loss 5
root@host# set thresholds total-loss 5
root@host# set destination-interface ge-0/0/1.0

root@host# edit services ip-monitoring policy IP-ROUTE
```

```
root@host# set match rpm-probe ISP1
root@host# set then preferred-route route 0.0.0.0/0 next-hop 2.2.1.254
root@host# set then preferred-route route 0.0.0.0/0 preferred-metric 5
```

Configure the security zone UNTRUST-1, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST-1
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/1.0 host-inbound-traffic system-services ping
```

Configure the security zone UNTRUST-2, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone UNTRUST-2
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ike
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/2.0 host-inbound-traffic system-services ping
```

Configure the security zone CLIENTS, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone CLIENTS
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/3.0 host-inbound-traffic system-services ping
```

Configure the security zone VOIP, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone VOIP
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ssh
root@host# set interfaces ge-0/0/4.0 host-inbound-traffic system-services ping
```

Configure the security zone ADVPN, assigning the interface to the zone plus allowing incoming administrative services:

```
root@host# edit security zones security-zone ADVPN
root@host# set interfaces st0.0 host-inbound-traffic protocols ospf
root@host# set interfaces st0.0 host-inbound-traffic system-services ping
root@host# set interfaces st0.1 host-inbound-traffic protocols ospf
root@host# set interfaces st0.1 host-inbound-traffic system-services ping
```

Configure SPI. Peers in a security association (SA) can become unsynchronized when one of the peers fails. For example, if one of the peers reboots, it might send an incorrect security parameter index (SPI). You can enable the device to detect such an event and resynchronize the peers by configuring the `bad-spi` response feature:

```
root@host# set security ike respond-bad-spi 5
```

Commit the changes:

```
root@host# commit
```

Configure the `router-id` used for OSPF:

```
root@ADVPN-SPOKE-1# set routing-options router-id 192.168.100.101
```

Configure the OSPF instance:

```
root@ADVPN-SPOKE-1# edit protocols ospf area 0.0.0.0
root@ADVPN-SPOKE-1# set interface st0.0 interface-type p2mp
root@ADVPN-SPOKE-1# set interface st0.0 metric 10
root@ADVPN-SPOKE-1# set interface st0.0 demand-circuit
root@ADVPN-SPOKE-1# set interface st0.0 flood-reduction
root@ADVPN-SPOKE-1# set interface st0.0 dynamic-neighbors
root@ADVPN-SPOKE-1# set interface st0.0 retransmit-interval 1
root@ADVPN-SPOKE-1# set interface st0.0 dead-interval 40
root@ADVPN-SPOKE-1# set interface st0.1 interface-type p2mp
root@ADVPN-SPOKE-1# set interface st0.1 metric 10
root@ADVPN-SPOKE-1# set interface st0.1 demand-circuit
root@ADVPN-SPOKE-1# set interface st0.1 flood-reduction
root@ADVPN-SPOKE-1# set interface st0.1 dynamic-neighbors
root@ADVPN-SPOKE-1# set interface st0.1 retransmit-interval 1
root@ADVPN-SPOKE-1# set interface st0.1 dead-interval 40
root@ADVPN-SPOKE-1# set interface ge-0/0/3.0 passive
root@ADVPN-SPOKE-1# set interface ge-0/0/4.0 passive
```

Now configure and request the certificates that you need to establish our tunnel. First define how to find the CA:

```
root@ADVPN-SPOKE-1# edit security pki ca-profile Our-CA_Server
root@ADVPN-SPOKE-1# set ca-identity MydomainCertificateAuthority
```

Now specify the CA SCEP URL:

```
root@ADVPN-SPOKE-1# set enrollment url http://1.1.1.2/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate (we have disabled the verification in this step):

```
root@ADVPN-SPOKE-1# set revocation-check use-ocsp ocsp
root@ADVPN-SPOKE-1# set revocation-check ocsp url http://1.1.1.2/ocsp
```

Keep in mind that the `challenge-password` is unique to your CA:

```
root@ADVPN-SPOKE-1# edit security pki auto-re-enrollment certificate-id ADVPN-PEER
root@ADVPN-SPOKE-1# set ca-profile-name Our-CA_Server
root@ADVPN-SPOKE-1# set re-generate-keypair
root@ADVPN-SPOKE-1# set re-enroll-trigger-time-percentage 10
root@ADVPN-SPOKE-1# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
root@ADVPN-SPOKE-1# set scep-encryption-algorithm des3
root@ADVPN-SPOKE-1# top
```

Activate this configuration and then verify IP connectivity before continuing:

```
root@ADVPN-SPOKE-1# commit
```

Verify that you can reach your gateway and also the remote gateway using an ICMP ping:

```
root@ADVPN-SPOKE-1# run ping 2.2.1.254
root@ADVPN-SPOKE-1# run ping 2.2.2.254
root@ADVPN-SPOKE-1# run ping 1.1.1.1
```

NOTE If you can't reach this gateway, please check your network and troubleshoot accordingly. When you have a working network, continue to the next step.

Let's define our address book objects that are used in the security policies:

```
root@ADVPN-SPOKE-1# set security address-book global address VOIP-Networks 21.20.0.0/16
root@ADVPN-SPOKE-1# set security address-book global address DATA-Networks 11.10.0.0/16
```

Now define a policy for each direction between VOIP and ADVPN:

```
root@ADVPN-SPOKE-1# edit security policies from-zone VOIP to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone VOIP policy 1
root@ADVPN-SPOKE-1# set match source-address VOIP-Networks
root@ADVPN-SPOKE-1# set match destination-address VOIP-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Define a policy for each direction between CLIENTS and ADVPN:

```
root@ADVPN-SPOKE-1# edit security policies from-zone CLIENTS to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone CLIENTS policy 1
root@ADVPN-SPOKE-1# set match source-address DATA-Networks
root@ADVPN-SPOKE-1# set match destination-address DATA-Networks
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close

root@ADVPN-SPOKE-1# edit security policies from-zone ADVPN to-zone ADVPN policy 1
root@ADVPN-SPOKE-1# set match source-address any
root@ADVPN-SPOKE-1# set match destination-address any
root@ADVPN-SPOKE-1# set match application any
root@ADVPN-SPOKE-1# set then permit
root@ADVPN-SPOKE-1# set then log session-init session-close
```

Because you need to be able to trace traffic that does not hit a firewall rule let's add a global config with a deny policy. There is no need to add the policy `default-policy` as long as you don't make any other changes to the global policy. For security precautions, you should also add this policy:

```
root@ADVPN-SPOKE-1# set security policies default-policy deny-all

root@ADVPN-SPOKE-1# edit security policies global
root@ADVPN-SPOKE-1# set policy 1 match source-address any
root@ADVPN-SPOKE-1# set policy 1 match destination-address any
root@ADVPN-SPOKE-1# set policy 1 match application any
root@ADVPN-SPOKE-1# set policy 1 then deny
root@ADVPN-SPOKE-1# set policy 1 then log session-init session-close
```

The next step is to enroll the root certificate from the trusted CA – if the fingerprint is okay, type `Yes`:

```
root@ADVPN-SPOKE-1# run request security pki ca-certificate enroll ca-profile Our-CA_Server
Fingerprint: (Your fingerprint will be unique)
 df:68:8d:7a:dd:5d:3d:2a:32:fc:27:e4:e0:dd:22:3e:81:51:44:b4 (sha1)
 a8:5e:be:0d:f2:f9:e3:83:fe:e8:05:d3:01:0f:1b:97 (md5)
Do you want to load the above CA certificate ? [yes,no] (Yes)

CA certificate for profile Our-CA_Server loaded successfully
```

Run the next command to see the current numbers of valid OCSP verifications:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now verify that you trust the downloaded certificate:

```
root@ADVPN-SPOKE-1# run request security pki ca-certificate verify ca-profile Our-CA_Server

CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
```

By running this command, you can see whether you have a successful verification or not. If the number is not updated then you have a problem with the certificate or the OCSP response service:

```
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

Now generate a `key-pair` to be used for the local certificate:

```
root@ADVPN-SPOKE-1# run request security pki generate-key-pair certificate-id ADVPN-PEER size 2048 type
rsa
```

When you enroll your local-certificate you need to add some input per device. Keep in mind that the challenge-password is unique to your CA:

```
root@ADVPN-SPOKE-1# run request security pki local-certificate enroll ca-profile Our-CA_Server
certificate-id ADVPN-PEER domain-name advpn-spoke-1.mydomain.com ip-address 2.2.1.1 subject
DC=mydomain.com,L=Stockholm,O=Mydomain,OU=LAB,CN=advpn-spokes-1 challenge-password
8CDB49EEEC84401A85D5F58800DB2F96
```

After you enroll for the `local-certificate` wait about a minute, re-run the `ocsp` command, and then you can verify that it's loaded and trusted:

```
root@ADVPn-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X

root@ADVPN-SPOKE-1# run request security pki local-certificate verify certificate-id ADVPN-PEER
CA certificate Our-CA_Server: Revocation check is in progress.Please check the PKId debug logs for
completion status
root@ADVPN-SPOKE-1# run show security pki statistics | match ocsp_rev_status
ocsp_rev_status_success:    X
ocsp_rev_status_revoked:    X
ocsp_rev_status_unknown:  X
```

IKE Proposal

At this stage, we should configure our IKE proposal. This can be used for multiple tunnels and by giving it a useful name, it's easier to troubleshoot if there is a need later on, which will be explained in the Troubleshooting section later in this chapter.

This proposal name tells you at a glance that it is authenticated with a Certificate and using a strong authentication and encryption algorithm with a rekey time of 28800sec:

```
root@ADVPN-SPOKE-1# edit security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set authentication-method rsa-signatures
root@ADVPN-SPOKE-1# set dh-group group14
root@ADVPN-SPOKE-1# set authentication-algorithm sha-256
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 28800
```

IKE Policy

In this example we use `ADVPN` as the name. If you had or were planning more AD-VPNs, then plan the name accordingly, for example, `ADVPN-Prim` and `ADVPN-Sec`:

```
root@ADVPN-SPOKE-1# edit security ike policy ADVPN
root@ADVPN-SPOKE-1# set mode main
root@ADVPN-SPOKE-1# set proposals CERT-DH14-SHA256-AES256-L28800
root@ADVPN-SPOKE-1# set certificate local-certificate ADVPN-SPOKE-1
root@ADVPN-SPOKE-1# set certificate peer-certificate-type x509-signature
```

IKE Gateway

```
root@ADVPN-SPOKE-1# edit security ike gateway ADVPN-HUB-PRI
root@ADVPN-SPOKE-1# set ike-policy ADVPN
root@ADVPN-SPOKE-1# set address 1.1.1.1
root@ADVPN-SPOKE-1# set address 1.1.2.1
root@ADVPN-SPOKE-1# set dead-peer-detection probe-idle-tunnel
root@ADVPN-SPOKE-1# set local-identity distinguished-name
```

```
root@ADVPN-SPOKE-1# set remote-identity distinguished-name container CN=advpn-hub
root@ADVPN-SPOKE-1# set external-interface ge-0/0/1.0
root@ADVPN-SPOKE-1# set advpn suggester disable
root@ADVPN-SPOKE-1# set advpn partner connection-limit 10
root@ADVPN-SPOKE-1# set advpn partner idle-time 60
root@ADVPN-SPOKE-1# set advpn partner idle-threshold 5
root@ADVPN-SPOKE-1# set version v2-only

root@ADVPN-SPOKE-1# edit security ike gateway ADVPN-HUB-SEC
root@ADVPN-SPOKE-1# set ike-policy ADVPN
root@ADVPN-SPOKE-1# set address 1.1.1.1
root@ADVPN-SPOKE-1# set address 1.1.2.1
root@ADVPN-SPOKE-1# set dead-peer-detection probe-idle-tunnel
root@ADVPN-SPOKE-1# set local-identity distinguished-name
root@ADVPN-SPOKE-1# set remote-identity distinguished-name container CN=advpn-hub
root@ADVPN-SPOKE-1# set external-interface ge-0/0/2.0
root@ADVPN-SPOKE-1# set advpn suggester disable
root@ADVPN-SPOKE-1# set advpn partner connection-limit 10
root@ADVPN-SPOKE-1# set advpn partner idle-time 60
root@ADVPN-SPOKE-1# set advpn partner idle-threshold 5
root@ADVPN-SPOKE-1# set version v2-only
```

IPsec Proposal

```
root@ADVPN-SPOKE-1# edit security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-SPOKE-1# set protocol esp
root@ADVPN-SPOKE-1# set authentication-algorithm hmac-sha-256-128
root@ADVPN-SPOKE-1# set encryption-algorithm aes-256-cbc
root@ADVPN-SPOKE-1# set lifetime-seconds 3600
```

IPsec Policy

```
root@ADVPN-SPOKE-1# edit security ipsec policy ADVPN
root@ADVPN-SPOKE-1# set perfect-forward-secrecy keys group14
root@ADVPN-SPOKE-1# set proposals ESP-SHA256-AES256-L3600
```

IPsec VPN

```
root@ADVPN-SPOKE-1# edit security ipsec vpn ADVPN-HUB-PRI
root@ADVPN-SPOKE-1# set bind-interface st0.0
root@ADVPN-SPOKE-1# set ike gateway ADVPN-HUB-PRI
root@ADVPN-SPOKE-1# set ike ipsec-policy ADVPN
root@ADVPN-SPOKE-1# set establish-tunnels immediately
```

Because you want to have any default route active for the external interface, you can set the "`establish-tunnels`" statement to "`on-traffic`" to reduce logs and resources from the device. It also lets traffic establish the tunnel when the primary link is down. (Be aware that this can result in application problems, if there is a need to reach resources behind the spoke from the central site all the time):

```
root@ADVPN-SPOKE-1# edit security ipsec vpn ADVPN-HUB-SEC
root@ADVPN-SPOKE-1# set bind-interface st0.1
root@ADVPN-SPOKE-1# set ike gateway ADVPN-HUB-SEC
root@ADVPN-SPOKE-1# set ike ipsec-policy ADVPN
root@ADVPN-SPOKE-1# set establish-tunnels immediately
```

And add syslog for troubleshooting:

```
root@ADVPN-SPOKE-1# set system syslog user * any emergency
root@ADVPN-SPOKE-1# edit system syslog
root@ADVPN-SPOKE-1# set file messages any any
root@ADVPN-SPOKE-1# set file messages authorization info
root@ADVPN-SPOKE-1# set file messages change-log none
root@ADVPN-SPOKE-1# set file messages interactive-commands none
root@ADVPN-SPOKE-1# set file messages structured-data
root@ADVPN-SPOKE-1# set file interactive-commands interactive-commands any
```

Commit the configuration:

```
root@ADVPN-SPOKE-1# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the troubleshooting section for ADVPN.

On the Hub

This should give you SAs for all the spokes. Index and cookies will change:

```
root@ADVPN-HUB# run show security ike security-associations
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
6139907    UP 164749a9f9f6d00a 4d5f206101f4088c IKEv2  2.2.1.1
6139908    UP 28c5fc6d4dca14ab 0ce428bdfde37a0d IKEv2  2.2.2.1
```

Verify that you see all your OSPF neighbors and have full state on all of them. The address statement shows the next-hop and the ID statement show the router ID from the `routing-options` stanza:

```
root@ADVPN-HUB# run show ospf neighbor
Address      Interface  State  ID   Pri Dead
192.168.100.102  st0.0    Full 192.168.100.102  128 -
192.168.100.101  st0.0    Full 192.168.100.101  128 -
```

When we see that you have a OSPF neighborship with all peers, you should verify that you received all routes and that each spoke has a different next-hop:

```
root@ADVPN-HUB# run show route protocol ospf
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

11.10.101.0/24   *[OSPF/10] 16:53:36, metric 11
> to 192.168.100.101 via st0.0
11.10.102.0/24   *[OSPF/10] 16:25:42, metric 11
> to 192.168.100.102 via st0.0
21.10.101.0/24   *[OSPF/10] 16:53:36, metric 11
> to 192.168.100.101 via st0.0
21.10.102.0/24   *[OSPF/10] 16:25:42, metric 11
> to 192.168.100.102 via st0.0
192.168.100.101/32 *[OSPF/10] 16:53:36, metric 10
```

```
> to 192.168.100.101 via st0.0
192.168.100.102/32 *[OSPF/10] 16:25:42, metric 10
> to 192.168.100.102 via st0.0
224.0.0.5/32   *[OSPF/10] 20:35:43, metric 1
MultiRecv
```

Verify active default route (active route is marked in boldface due to lower metric). If you not have your primary default gateway up, go to the troubleshooting section:

```
root@ADVPN-HUB# run show route 0.0.0.0
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0  *[Static/5] 17:24:19, metric 10
> to 1.1.1.254 via ge-0/0/1.0
 [Static/5] 18:42:51, metric 11
> to 1.1.2.254 via ge-0/0/2.0
```

On Spokes

This should give you the hub SAs with the primary link up on the hub site. Index and cookies will change.

```
root@ADVPN-SPOKE-1# run show security ike security-associations
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275816    UP 164749a9f9f6d00a 4d5f206101f4088c IKEv2  1.1.1.1
```

Verify that we have full neighborship with the hub:

```
root@ADVPN-SPOKE-1# run show ospf neighbor
Address      Interface  State  ID  Pri Dead
192.168.100.254  st0.0    Full 192.168.100.254  128 -
```

When we see that you have neighborship with the hub, you should verify that you receives all routes and that they are pointing to the hub:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

11.10.0.0/24   *[OSPF/10] 16:55:00, metric 11
> to 192.168.100.254 via st0.0
11.10.102.0/24   *[OSPF/10] 16:27:06, metric 21
> to 192.168.100.254 via st0.0
21.10.0.0/24   *[OSPF/10] 16:55:00, metric 11
> to 192.168.100.254 via st0.0
21.10.102.0/24   *[OSPF/10] 16:27:06, metric 21
> to 192.168.100.254 via st0.0
192.168.100.102/32 *[OSPF/10] 16:27:06, metric 20
> to 192.168.100.254 via st0.0
192.168.100.253/32 *[OSPF/10] 16:55:00, metric 10
> to 192.168.100.254 via st0.0
192.168.100.254/32 *[OSPF/10] 16:55:00, metric 10
> to 192.168.100.254 via st0.0
224.0.0.5/32   *[OSPF/10] 19:56:12, metric 1
MultiRecv
```

Verify the active default route (active route is marked in boldface due to lower metric). If you not have your primary default gateway up, go to the troubleshooting section:

```
root@ADVPN-HUB# run show route 0.0.0.0
inet.0: 20 destinations, 23 routes (20 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0  *[Static/5] 17:24:19, metric 10
> to 2.2.1.254 via ge-0/0/1.0
 [Static/5] 18:42:51, metric 11
> to 3.3.1.254 via ge-0/0/2.0
```

When you have traffic between the spokes, this should give you the shortcut SAs. Index and Cookies will change. You will see the different active shortcut tunnels with the "**Remote Address**":

```
root@ADVPN-SPOKE-1# run show security ike security-associations sa-type shortcut
Index      State  Initiator cookie  Responder cookie  Mode Remote Address
3275817    UP c47ecb3ac420b588 e0f48944fb66b456 IKEv2 2.2.2.1
```

When you have an active shortcut tunnel, you can see that you now have another path to the destination as compared to before the shortcut tunnel. You can see this as the next-hop has changed:

```
root@ADVPN-SPOKE-1# run show route protocol ospf
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

11.10.0.0/24   *[OSPF/10] 16:55:48, metric 11
> to 192.168.100.254 via st0.0
11.10.102.0/24   *[OSPF/10] 00:00:05, metric 11
> to 192.168.100.102 via st0.0
21.10.0.0/24   *[OSPF/10] 16:55:48, metric 11
> to 192.168.100.254 via st0.0
21.10.102.0/24   *[OSPF/10] 00:00:05, metric 11
> to 192.168.100.102 via st0.0
192.168.100.102/32 *[OSPF/10] 00:00:05, metric 10
> to 192.168.100.102 via st0.0
192.168.100.253/32 *[OSPF/10] 16:55:48, metric 10
> to 192.168.100.254 via st0.0
192.168.100.254/32 *[OSPF/10] 16:55:48, metric 10
> to 192.168.100.254 via st0.0
224.0.0.5/32   *[OSPF/10] 19:57:00, metric 1
```

Troubleshooting

There are three sections here that can help you troubleshoot the following issues:

- No-established tunnel
- Connectivity issues
- Redundancy issues

No-established Tunnel

Text that has boldface indicates the problem, text below each output describes the problem or how to proceed.

No-established tunnel means that you are actively trying to establish a tunnel, but you did not receive any response from the remote peer. This could be a connectivity issue, such as routing or another networking problem, but it could also be a configuration mistake on the remote peer:

```
root@ADVPN-SPOKE-1# run show sec ipsec inactive-tunnels detail
ID: 67108866 Virtual-system: root, VPN Name: ADVPN-HUB
Local Gateway: 2.2.1.1, Remote Gateway: 1.1.1.1
Local Identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Remote Identity: ipv4_subnet(any:0,[0..7]=0.0.0.0/0)
Version: IKEv2
DF-bit: clear, Bind-interface: st0.0
Port: 500, Nego#: 0, Fail#: 0, Def-Del#: 0 Flag: 0x8608a29
Tunnel events:
Wed Jan 14 2015 11:54:01 -0800: IKE exchange is in progress currently (1 times)
Wed Jan 14 2015 11:53:20 -0800: No response from peer. Negotiation failed (2 times)
Wed Jan 14 2015 11:04:16 -0800: Tunnel is ready. Waiting for trigger event or peer to trigger
negotiation (1 times)
```

If there is no other information, troubleshoot the local device:

```
root@ADVPN-HUB# run ping 1.1.1.1
```

If you get a response from the peer but negotiation fails, continue your troubleshooting on the peer; if not, check your network connectivity.

On the hub:

```
root@ADVPN-HUB# run show security flow session destination-port 500
Session ID: 35059, Policy name: self-traffic-policy/1, Timeout: 60, Valid
In: 2.2.1.1/500 --> 1.1.1.1/500;udp, If: .local..0, Pkts: 188, Bytes: 64148
Out: 1.1.1.1/500 --> 2.2.1.1/500;udp, If: ge-0/0/1.0, Pkts: 0, Bytes: 0
```

With this output, you can see the incoming IKE packet from our advpn-spoke-1, but there is no response from the hub. If you do not find any session at all, you know that advpn-spoke-1 can't reach this hub with IKE even if pings reach the hub. If you see an incoming session, go to the next step.

Check that you have IKE configured and allowed on the external interface. Otherwise, add the following:

```
root@ADVPN-HUB# show security zones security-zone UNTRUST
interfaces {
 ge-0/0/1.0 {
 host-inbound-traffic {
  system-services {
  ike;
  ssh;
  ping;
  }
 }
 }
}
```

Verify that both the local and CA certificates are loaded on the device using the commands below:

```
root@ADVPN-HUB-1# run show security pki ca-certificate
root@ADVPN-HUB-1# run show security pki local-certificate
```

If both certificates are loaded on the box, verify them using the commands below. Otherwise retrieve the missing certificates and verify them when they are loaded.

Verify the certificates using the commands here:

```
root@ADVPN-HUB# run request security pki ca-certificate verify ca-profile Our-CA_Server
root@ADVPN-HUB# run request security pki local-certificate verify certificate-id ADVPN-HUB
CA certificate Our-CA_Server verified successfully
```

If the certificates are not verified successfully, it may be that you can't reach the CRL/OCSP server or you have forgotten to disable the revocation check.

Next step is to check that you have the right IKE configuration. Check the following:

- Wildcard = is matching the remote peer's local-certificate
- local-identity = distinguished-name
- External-interface = correct external interface
- Partner = is in disable state
- Version = Should be v2-only not v1

```
root@ADVPN-HUB# show security ike gateway ADVPN-SPOKES
ike-policy ADVPN;
dynamic {
 distinguished-name {
 wildcard OU=Branch;
 }
 ike-user-type group-ike-id;
}
```

```
local-identity distinguished-name;
external-interface ge-0/0/1.0;
advpn {
 partner {
 disable;
 }
}
version v2-only;
```

Verify that you have the right secure tunnel interface bound to this tunnel:

```
root@ADVPN-HUB# show security ipsec vpn ADVPN-SPOKES
bind-interface st0.0;
ike {
 gateway ADVPN-SPOKES;
 ipsec-policy ADVPN;
}
```

Verify that you have configured the ST0.x interface both as multipoint and with a subnet that matches the remote peer's ST interface:

```
root@ADVPN-HUB# show interfaces st0 unit 0
description advpn;
multipoint;
family inet {
 address 192.168.100.254/24;
}
```

Verify that both your IKE and IPsec proposals and policies match the remote side.

```
root@ADVPN-HUB-1# show security ike proposal CERT-DH14-SHA256-AES256-L28800
root@ADVPN-HUB-1# show security ike policy ADVPN
root@ADVPN-HUB-1# show security ipsec proposal ESP-SHA256-AES256-L3600
root@ADVPN-HUB-1# show security ipsec policy ADVPN
```

If you haven't found any configuration mistake at this stage, then enable traceoptions or reconfigure all steps in the guide:

```
root@ADVPN-HUB# run request security ike debug-enable local 1.1.1.1 remote 2.2.1.1
```

Check the log file and try to figure out what could cause the problem:

```
run show log kmd
```

Connectivity Issues

First you should verify that you learned all routes through OSPF from the branch devices. You should now see IP prefixes from all ge-0/0/3.0, ge-0/0/4.0 interfaces on all connected devices. If you are missing routes, you probably missed adding the corresponding interface under your OSPF configuration at the remote peer. If you are missing all routes from one peer, check your OSPF configuration. (See steps below.) If you see that you have fewer active routes than learned, go to the next step. One route will always be hidden, and one less active than the total of destinations:

```
root@ADVPN-HUB# run show route protocol ospf
inet.0: 53 destinations, 53 routes (52 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

ex below
11.10.101.0/24 *[OSPF/10] 15:18:17, metric 11
  > to 192.168.100.101 via st0.0
21.10.101.0/24 *[OSPF/10] 15:18:17, metric 11
  > to 192.168.100.101 via st0.0
192.168.100.101/32 *[OSPF/10] 15:18:17, metric 10
  > to 192.168.100.101 via st0.0
224.0.0.5/32 *[OSPF/10] 17:02:44, metric 1
MultiRecv
```

Check to see if you have any inactive-path for your prefix. You most likely have configured a routing policy that makes this route inactive:

```
root@ADVPN-HUB# run show route inactive-path
```

If you are missing a specific route, check the OSPF configuration for all the below statements. And verify that all interfaces are in your configuration:

```
root@ADVPN-SPOKE-1# show protocols ospf
area 0.0.0.0 {
 interface st0.0 {
 interface-type p2mp;
 metric 10;
 demand-circuit;
 flood-reduction;
 dynamic-neighbors;
 }
 interface ge-0/0/3.0 {
 passive;
 }
 interface ge-0/0/4.0 {
 passive;
 }
}
```

If you are still missing any prefix, check that the corresponding interface for your missing prefix is active and up:

```
root@ADVPN-SPOKE-1# run show interfaces terse
```

If you are missing all routes from one peer, check the neighbor status:

```
root@ADVPN-HUB-1# run show ospf neighbor
```

If you do not get full state after a minute, enable traceoptions to troubleshoot further:

```
root@ADVPN-HUB-1# set protocols ospf traceoptions file OSPF
root@ADVPN-HUB-1# set protocols ospf traceoptions file size 5m
root@ADVPN-HUB-1# set protocols ospf traceoptions flag all
root@ADVPN-HUB-1# commit
```

Then check your log file to find out what's causing the problem:

```
root@ADVPN-HUB-1# run show log OSPF
```

Redundancy Issues

If you see the remote peer with a non-primary tunnel, check the following:

```
root@ADVPN-HUB# run show services ip-monitoring status
Policy - IP-ROUTE (Status: FAIL)
RPM Probes:
Probe name   Test Name  Address    Status
-------------------- ---------------  ---------------- ---------
ISP1           IP-ROUTE 1.1.1.253  FAIL

Route-Action:
route-instance   route    next-hop state
-----------------  --------------------------------- -------------
inet.0         0.0.0.0/0  1.1.2.254  APPLIED
```

This means you can't reach the IP-probe 1.1.1.253 using the primary interface and you have failed on the secondary interface.

To find out when you lost the primary connection, you can use the command below. When you don't see any 'Round trip time' you can't reach the IP probe address:

```
root@ADVPN-HUB# run show services rpm history-results
Owner,     Test   Probe received     Round trip time
ISP1,      IP-ROUTE Thu Jan 29 09:34:48 2015   1694 usec
ISP1,      IP-ROUTE Thu Jan 29 09:34:50 2015   1738 usec
ISP1,      IP-ROUTE Thu Jan 29 09:34:52 2015   1611 usec
```

GroupVPNv2

Group VPN, also known as Group Domain of Interpretation (GDOI), is a technology to build encrypted connectivity between your data center and all remote locations without the need to manually configure each site as a new location is added or removed. Instead, all configuration is done at the central site and then distributed to each of the members in a secure and encrypted fashion.

There's a slight difference between standard based IPsec VPN, because the purpose of Group VPNs is to build full-mesh VPNs instead of Site-to-Site VPNs, which is the prime purpose of traditional IPsec VPNs. To do this, you need a Central Group Key Server (KS) and Group Members (GM), that is, the local gateway for each location. You could explain this as the Key server is the parent, while the group members are children following the parent's decisions.

All together this makes the solution easier to manage, as smaller boxes can be used as only one or a few IPsec SAs will be used. This is no burden when it comes to CoS or dynamic routing protocols. Keep in mind that this topology can not to be used over the Internet, as you must preserve the IP header.

NOTE GroupVPNv2 is supported on all SRX Series platforms except for the SRX5xxx Series.

GroupVPNv2 Deployment with up to 2000 GMs

In this Group VPN use case, there are three locations that should have IPsec encrypted traffic between them. Because this is a deployment with less then 2000 GMs, you can use a single KS to accomplish the task. Because you want the KS to be redundant, you should form a SRX Chassis cluster for redundancy.

If you want to add more members, that will be explained in the example config. We will also show the difference of configuration between the SRX Series and the MX Series.

NOTE Configuration is displayed in snippets below. Keep in mind that you need to change variable data such as interface, IP address, and other information related to your own network.

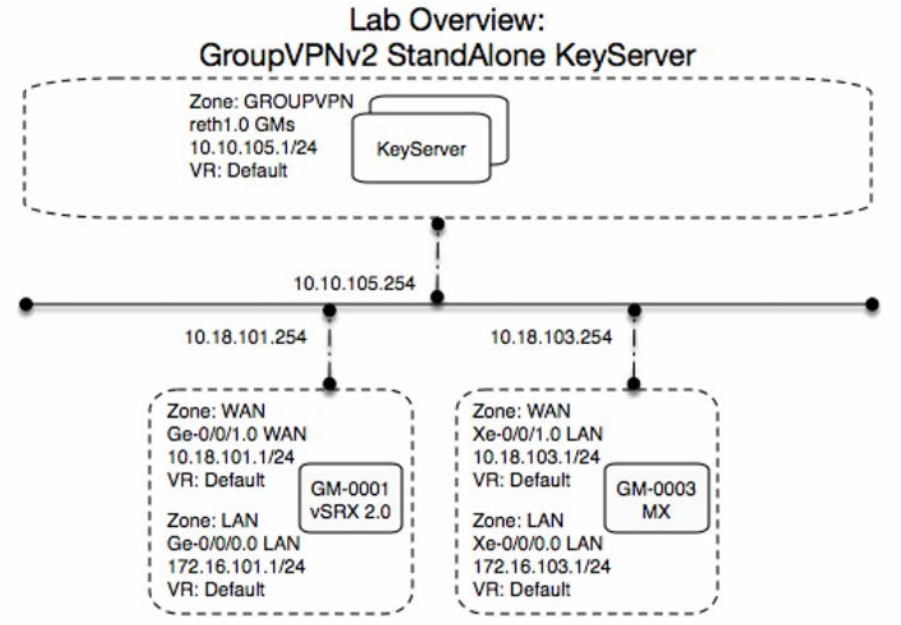


Figure 4.4 GroupVPN2 StandAlone KeyServer

Requirements

Hardware: GC/KS - Juniper vSRX 2.0

GM - Juniper vSRX 2.0

GM - Juniper MX Routers using MS-MIC/MPC

Software: vSRX - Junos 15.1X49D30

MX - Junos 15.1r2

Routing: It's highly recommended to use a dynamic routing topology between all GroupMembers and GS/KS, this is not described.

NTP timing: All devices should be configured for NTP timing.

Configuration

This step-by-step cookbook demands that you have a working IP routing topology that allows connectivity between all of your sites. Before starting, configure a security policy that allows all outgoing and incoming traffic between the WAN and LAN zones on all group members to verify your routing topology. You should be able to send traffic from a host on each LAN segment to a host on any of the other LAN segments. When this has been verified, please continue with the cookbook, and you will change and update all configuration under the security stanza on SRX related devices. As MX does not use the security stanza, you need to make any changes to that. Keep in mind that you might experience traffic interruption when taking this configuration active the first time. You will also change the hostname on all devices to make it easier to follow the cookbook.

Configuring the Key Server

It's highly recommended to form two SRX devices into a SRX chassis cluster for redundancy. This process is not described in this cookbook, for a description please consult the Juniper TechLibrary at: https://www.juniper.net/documentation/en_US/junos/topics/task/operational/chassis-cluster-srx-series-creating.html.

First delete all current configurations under the security stanza:

```
root@host# delete security
```

Change the hostname to make it easier to troubleshoot (not mandatory):

```
root@host# set system host-name KeyServer
```

Set system clock to current state before enabling NTP:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Now configure an IKE proposal to negotiate IKE:

```
root@host# edit security group-vpn server ike proposal PSK-SHA256-DH14-AES256
root@host# set authentication-method pre-shared-keys
root@host# set authentication-algorithm sha-256
root@host# set dh-group group14
root@host# set encryption-algorithm aes-256-cbc
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to each remote GM. You should also configure the pre-shared key with a strong one. A weak one is used in this example:

```
root@host# edit security group-vpn server ike policy GMs
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 1234567890
root@host# top
```

As a next step, you should now define all your GMs. You will only configure three GMs, and if you need more in your topology, please just repeat these steps for each GM. We will use GM-0001, and GM-0003 in this chapter to cover how this is done on an SRX and a MX Series device.

```
root@host# edit security group-vpn server ike gateway GM-0001
root@host# set ike-policy GMs
root@host# set address 10.18.101.1
root@host# set local-address 10.10.105.1
root@host# top edit security group-vpn server ike gateway GM-0003

root@host# set ike-policy GMs
root@host# set address 10.18.103.1
root@host# set local-address 10.10.105.1
root@host# top edit security group-vpn server ike gateway GM-0005

root@host# set ike-policy GMs
root@host# set address 10.18.105.1
root@host# set local-address 10.10.105.1
root@host# top
```

Next configure the IPsec proposal; that defines how you will encrypt the traffic:

```
root@host# edit security group-vpn server ipsec proposal AES256-SHA256-L3600
root@host# set authentication-algorithm hmac-sha-256-128
root@host# set encryption-algorithm aes-256-cbc
root@host# set lifetime-seconds 3600
root@host# top
```

Now it's time to configure a group-identifier that will define which encryption policy we will distribute to the GMs that have authenticated themselves for this group. Each step will be described.

First define a logical name for each group that you configure. Most users have a few groups that will define which GMs can share encrypted traffic between them. This is just to give you an understating of how it works. We choose Group_ID-0001 and will use the identifier 1:

```
root@host# edit security group-vpn server group GROUP_ID-0001
```

This is the Group-identifier:

```
root@host# set group-id 1
```

Now define how many GMs that could subsribe to this Group:

```
root@host# set member-threshold 2000
```

Let's bind each GM from the IKE configuration above to this Group:

```
root@host# set ike-gateway GM-0001
root@host# set ike-gateway GM-0003
root@host# set ike-gateway GM-0005
```

Anti-replay is a good way to prevent spoofed packets, so that's why this should be as low as possible. Keep in mind that a GM hosted in a virtual environment might require a higher value than a physical machine, as the host system might process packets more slowly, meaning packets might come out of sync. This value must be high enough to work for all devices, if used. Let's try it with a value of 1000 milliseconds:

```
root@host# set anti-replay-time-window 1000
```

Next define how what security protection to use when the KeyServer updates the GMs.

Here let's use PUSH notification to the GMs using unicast:

```
root@host# edit server-member-communication
root@host# set communication-type unicast
root@host# set lifetime-seconds 7200
root@host# set encryption-algorithm aes-256-cbc
root@host# set sig-hash-algorithm sha-256
root@host# up
```

Now define the IPsec proposal configured above, but also define which networks should be encrypted and in what direction. Because you want to allow all LAN segments to exchange encrypted traffic, use 172.16.0.0/12 for both source and destination to minimize configuration:

```
root@host# edit ipsec-sa GROUP_ID-0001
root@host# set proposal AES256-SHA256-L3600
root@host# set match-policy 1 source 172.16.0.0/12
root@host# set match-policy 1 destination 172.16.0.0/12
root@host# set match-policy 1 protocol 0
root@host# top
```

Because you deleted all the configurations under security, you need to define a few security policies.

First, define the default policy and then a global policy that will act as a clean up policy to log any traffic missing the security policy for logging purposes. Keep in mind that the reject option for the global policy is recommended to change to deny for a real deployment. We use reject in this case as that will make troubleshooting faster:

```
root@host# set security policies default-policy deny-all

root@host# edit security policies global policy 1000
root@host# set match source-address any
root@host# set match destination-address any
root@host# set match application any
root@host# set match from-zone any
root@host# set match to-zone any
root@host# set then reject
root@host# set then log session-init
root@host# set then count
root@host# top
```

Finally you must configure your interface to be included into each respective security zone. If you have more then one interface on your KS, don't forget to reconfigure that one so it works as you want. Because this cookbook only has one interface for the KS, it also allows incoming IKE and SSH traffic: IKE for the GM to connect and SSH for management.. Keep in mind that we use a `reth` interface here because we expect that you have a SRX chassis cluster, if you do not, change the interface to meet your environment:

```
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ike
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ssh
root@host# set security zones security-zone GROUPVPN interfaces reth1x<.0
```

Commit this configuration and let's continue with our Group Members:

```
root@host# commit
```

GroupVPNv2 - Deployment with More than 2000 GMs

In this use case, there are three locations that should have IPsec-encrypted traffic between them. Because this is a deployment with more then 2000 GMs, you can use a server-cluster deployment. The root-server should be configured as a SRX chassis cluster for redundancy. Depending on the amount of GMs, you would need an appropriate amount of sub-servers. Please reference the technical documentation for scaling, but this section will configure four sub-servers to give you maximum scale.

NOTE Configuration is displayed in snippets below. Keep in mind that you need to change variable data such as interface, IP address, and other information related to your own network.

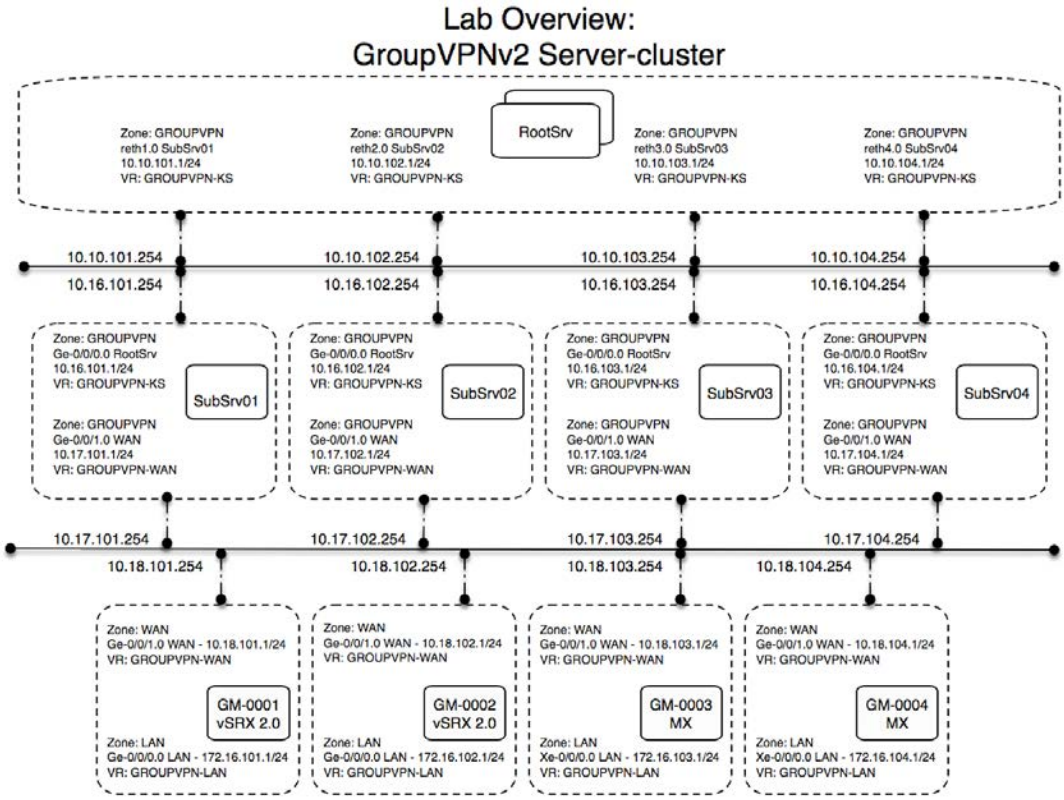


Figure 4.5 GroupVPNv2 Server-Cluster

Requirements

Hardware:

GC/KS - Juniper vSRX 2.0

GM - Juniper vSRX 2.0

GM - Juniper MX Routers using MS-MIC/MPC

vSRX - Junos 15.1X49D30

MX - Junos 15.1r2

Routing: It's highly recommended to use a dynamic routing topology between all Group Members and GS/KS, but this is not covered.

NTP timing: All devices should be configured for NTP timing.

Configuration

This step-by-step book demands that you have a working IP routing topology that allows connectivity between all of your sites. Before starting, configure a security policy that allows all outgoing and incoming traffic between the WAN and LAN zones on all group members to verify your routing topology. You should be able to send traffic from a host on each LAN segment to a host on any of ther other LAN segments. When this has been verified. Please continue with this book and we will change and update all configuration under the security stanza on SRX related devices. As MX does not use the security stanza, we would need to make any changes to that. Keep in mind that you might experience traffic interruption when taking this configuration active the first time. We will also change the hostname on all devices to make it easier to follow the book.

Configuring the Root Server

It's highly recommended to form two SRX devices into a SRX chassis cluster for redundancy. This is not described in this book, but it is within the Juniper TechLibrary at: https://www.juniper.net/documentation/en_US/junos/topics/task/operational/chassis-cluster-srx-series-creating.html.

First, delete all current configurations under the security stanza:

```
root@host# delete security
```

Change the hostname to make it easier to troubleshoot (not mandatory):

```
root@host# set system host-name RootSrv
```

Set system clock to current state before enabling NTP:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
root@host# top
```

Now configure an IKE proposal to negotiate IKE:

```
root@host# edit security group-vpn server ike proposal PSK-SHA256-DH14-AES256
root@host# set authentication-method pre-shared-keys
root@host# set authentication-algorithm sha-256
root@host# set dh-group group14
root@host# set encryption-algorithm aes-256-cbc
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to each remote sub-server. You should also configure the pre-shared key with a strong one. A weak one is used in this example:

```
root@host# edit security group-vpn server ike policy SubSrv
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 0987654321
root@host# top
```

As a next step, you should now define all your sub-servers:

```
root@host# edit security group-vpn server ike gateway SubSrv01
root@host# set ike-policy SubSrv
root@host# set dead-peer-detection always-send
root@host# set address 10.16.101.1
root@host# set local-address 10.10.101.1
root@host# top edit security group-vpn server ike gateway SubSrv02
root@host# set ike-policy SubSrv
root@host# set dead-peer-detection always-send
root@host# set address 10.16.102.1
root@host# set local-address 10.10.102.1
root@host# top edit security group-vpn server ike gateway SubSrv03
root@host# set ike-policy SubSrv
root@host# set dead-peer-detection always-send
root@host# set address 10.16.103.1
root@host# set local-address 10.10.103.1
root@host# top edit security group-vpn server ike gateway SubSrv04
root@host# set ike-policy SubSrv
root@host# set dead-peer-detection always-send
root@host# set address 10.16.104.1
root@host# set local-address 10.10.104.1
root@host# top
```

Next configure the IPsec proposal, that defines how you will encrypt the traffic:

```
root@host# edit security group-vpn server ipsec proposal AES256-SHA256-L3600
root@host# set authentication-algorithm hmac-sha-256-128
root@host# set encryption-algorithm aes-256-cbc
root@host# set lifetime-seconds 3600
root@host# top
```

Now it's time to configure a group identifier that will define which encryption policy we will distribute to sub-server and GM that have authenticated themselves for this group. Each step will be described.

First define a logical name for each group that you configure. Most users have a few groups that will define which GMs can share encrypted traffic between them. This is just to give you an understating of how it works. We choose Group_ID-0001 and will use the identifier 1:

```
root@host# edit security group-vpn server group GROUP_ID-0001
```

This is the group identifier:

```
root@host# set group-id 1
```

Now define how many GMs could subsribe to this Group:

```
root@host# set member-threshold 2000
```

Define the server-cluster functions:

```
root@host# edit server-cluster
```

Define this host as the root-server:

```
root@host# set server-role root-server
```

Let's bind each sub-server from the IKE configuration above to this group:

```
root@host# set ike-gateway SubSrv01
root@host# set ike-gateway SubSrv02
root@host# set ike-gateway SubSrv03
root@host# set ike-gateway SubSrv04
```

Here we should also define the time period the root and sub-servers should retransmit in case of a problem:

```
root@host# set retransmission-period 10
root@host# up
```

Anti-replay is a good way to prevent spoofed packets, so that's why this should be as low as possible. Keep in mind that a GM hosted in a virtual environment might require a higher vaule then a physical machine as the host system might process packets more slowly, meaning packets might come out of sync. If used, this vaule must be high enough to work for all devices. Let's try it with a vaule of 1000 milliseconds:

```
root@host# set anti-replay-time-window 1000
```

Next define how what security protection to use when the KeyServer updates the GMs.

Here, let's use PUSH notification to the GMs using unicast:

```
root@host# edit server-member-communication
root@host# set communication-type unicast
root@host# set lifetime-seconds 7200
root@host# set encryption-algorithm aes-256-cbc
root@host# set sig-hash-algorithm sha-256
root@host# up
```

Now define the IPsec proposal configured above, but also define which networks should be encrypted and in what direction. Because you want to allow all LAN segments to exchange encrypted traffic, use 172.16.0.0/12 for both source and destination to minimize configuration:

```
root@host# edit ipsec-sa GROUP_ID-0001
root@host# set proposal AES256-SHA256-L3600
root@host# set match-policy 1 source 172.16.0.0/12
root@host# set match-policy 1 destination 172.16.0.0/12
root@host# set match-policy 1 protocol 0
root@host# top
```

Because you deleted all the configurations under security, you need to define a few security policies.

First, define the default policy and then a global policy that will act as a clean up policy to log any traffic missing the security policy for logging purposes. Keep in mind that the reject option for the global policy is recommended to change to deny for a real deployment. We use reject in this case as that will make troubleshooting faster:

```
root@host# set security policies default-policy deny-all

root@host# edit security policies global policy 1000
root@host# set match source-address any
root@host# set match destination-address any
root@host# set match application any
root@host# set match from-zone any
root@host# set match to-zone any
root@host# set then reject
root@host# set then log session-init
root@host# set then count
root@host# top
```

Finally, you must configure your interface to be included in each respective security zone. If you have more than one interface on your KS, don't forget to reconfigure that one so it works as you want. Because this book only has one interface for the KS, it also allows incoming IKE and SSH traffic: IKE for the GM to connect and SSH for management. Keep in mind that we use a `reth` interface here because we expect that you have a SRX chassis cluster, if you do not, change the interface to meet your environment:

```
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ike
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ssh
root@host# set security zones security-zone GROUPVPN interfaces reth1.0
root@host# set security zones security-zone GROUPVPN interfaces reth2.0
root@host# set security zones security-zone GROUPVPN interfaces reth3.0
root@host# set security zones security-zone GROUPVPN interfaces reth4.0
```

Commit this configuration and let's continue with the sub servers:

```
root@host# commit
```

Configuring the Sub-Servers

Sub-servers should not be configured as SRX chassis clusters.

First delete all current configuration under the security stanza:

```
root@host# delete security
```

Change the hostname to make it easier to troubleshoot (not mandatory):

```
root@host# set system host-name SubSrv01
```

Set system clock to current state before enabling NTP:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host#edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Now configure an IKE proposal to negotiate IKE:

```
root@host# edit security group-vpn server ike proposal PSK-SHA256-DH14-AES256
root@host# set authentication-method pre-shared-keys
root@host# set authentication-algorithm sha-256
root@host# set dh-group group14
root@host# set encryption-algorithm aes-256-cbc
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to each remote GM. You should also configure the pre-shared key with a strong one. A weak one is used in this example. Keep in mind it should be the same as on the root-server:

```
root@host# edit security group-vpn server ike policy RootSrv
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 0987654321
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to each Group member.

You should also configure the pre-shared key with a strong one. A weak one is used in this example. Keep in mind it should be the same as on the root server.

This step is not mandatory; you can use the previous policy with the same pre-shared key, but we prefer to have different ones between the root-server and group members:

```
root@host# edit security group-vpn server ike policy GMs
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 1234567890
root@host# top
```

Next define the root server. Keep an eye on the x – that should be the correct peer sub-server, see your topology:

```
root@host# edit security group-vpn server ike gateway RootSrv
root@host# set ike-policy RootSrv
root@host# set dead-peer-detection always-send
root@host# set address 10.10.10x.1
root@host# set local-address 10.16.10x.1
```

Now define all your Group members. Remember that this needs to be done on all sub-servers, both now and in the future, if you add more group members later on, and that the local addresses should be updated for each sub-server:

```
root@host# top edit security group-vpn server ike gateway GM-0001
root@host# set ike-policy GMs
root@host# set address 10.18.101.1
root@host# set local-address 10.17.10x.1
root@host# top edit security group-vpn server ike gateway GM-0003
root@host# set ike-policy GMs
root@host# set address 10.18.103.1
root@host# set local-address 10.17.10x.1
root@host# top edit security group-vpn server ike gateway GM-0005
root@host# set ike-policy GMs
root@host# set address 10.18.105.1
root@host# set local-address 10.17.10x.1
root@host# top
```

Next configure the IPsec proposal, that defines how you will encrypt the traffic:

```
root@host# edit security group-vpn server ipsec proposal AES256-SHA256-L3600
root@host# set authentication-algorithm hmac-sha-256-128
root@host# set encryption-algorithm aes-256-cbc
root@host# set lifetime-seconds 3600
root@host# top
```

Now it's time to configure a group-identifier that will define which encryption policy we will distribute to the GMs that have authenticated themselves for this group. Each step will be described.

First define a logical name for each group that you configure. Most users have a few groups that will define which GMs can share encrypted traffic between them. This is just to give you an understating of how it works. We choose Group_ID-0001 and will use the identifier 1:

```
root@host# edit security group-vpn server group GROUP_ID-0001
```

This is the Group-identifier:

```
root@host# set group-id 1
```

Now define how many GMs that could subsribe to this Group:

```
root@host# set member-threshold 2000
```

Let's bind each GM from the IKE configuration above to this Group:

```
root@host# set ike-gateway GM-0001
root@host# set ike-gateway GM-0003
root@host# set ike-gateway GM-0005
```

Now define this host as the sub-server:

```
root@host# edit server-cluster
root@host# set server-role sub-server
```

You also have to define the gateway to the root server:

```
root@host# set ike-gateway RootSrv
```

Here define the time period that the root and sub-servers should re-transmit in case of a problem:

```
root@host# set retransmission-period 10
root@host# up
```

Anti-replay is a good way to prevent spoofed packets, so that's why this should be as low as possible. Keep in mind that a GM hosted in a virtual environment might require a higher value then a physical machine, as the host system might process packets slower, meaning the packet might come out of sync. If used, this vaule must be high enough to work for all devices. Let's try it with a vaule of 1000 milliseconds:

```
root@host# set anti-replay-time-window 1000
```

Next define what security protection to use when the KeyServer updates the GMs.

Here let's use PUSH notification to the GMs using unicast:

```
root@host# edit server-member-communication
root@host# set communication-type unicast
root@host# set lifetime-seconds 7200
root@host# set encryption-algorithm aes-256-cbc
root@host# set sig-hash-algorithm sha-256
root@host# up
```

Now define the IPsec proposal configured above, but also define which networks should be encrypted and in what direction. Because you want to allow all LAN segments to exchange encrypted traffic, use 172.16.0.0/12 for both source and destination to minimize configuration:

```
root@host# edit ipsec-sa GROUP_ID-0001
root@host# set proposal AES256-SHA256-L3600
root@host# set match-policy 1 source 172.16.0.0/12
root@host# set match-policy 1 destination 172.16.0.0/12
root@host# set match-policy 1 protocol 0
root@host# top
```

Because you deleted all the configurations under security, you need to define a few security policies.

First, define the default policy and then a global policy that will act as a clean up policy to log any traffic missing the security policy for logging purposes. Keep in mind that the reject option for the global policy is recommended to change to deny for a real deployment. We use reject in this case as that will make troubleshooting faster:

```
root@host# set security policies default-policy deny-all

root@host# edit security policies global policy 1000
root@host# set match source-address any
root@host# set match destination-address any
root@host# set match application any
root@host# set match from-zone any
root@host# set match to-zone any
root@host# set then reject
root@host# set then log session-init
root@host# set then count
root@host# top
```

Finally you must configure your interface to be included in each respective security zone. If you have more than one interface on your KS, don't forget to reconfigure that one so it works as you want. Because this book only has one interface for the KS, it also allows incoming IKE and SSH traffic: IKE for the GM to connect and SSH for management. Keep in mind that we use a `reth` interface here because we expect that you have a SRX chassis cluster, if you do not, change the interface to meet your environment:

```
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ike
root@host# set security zones security-zone GROUPVPN host-inbound-traffic system-services ssh
root@host# set security zones security-zone GROUPVPN interfaces ge-0/0/0.0
root@host# set security zones security-zone GROUPVPN interfaces ge-0/0/1.0
```

Commit this configuration and let's continue with our sub-servers:

```
root@host# commit
```

Repeat this configuration for the three sub-servers, keeping in mind that you need to change the IP addresses to match your topology as well as your hostname.

Group Members

To configure the GMs, start with the SRX:

First delete all current configurations under the security stanza:

```
root@host# delete security
```

Change the hostname to make it easier to troubleshoot (not mandatory):

```
root@host# set system host-name GM-0001
```

Set system clock to current state before enabling NTP:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# edit system ntp
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Now configure an IKE proposal to negotiate IKE:

```
root@host# edit security group-vpn member ike proposal PSK-SHA256-DH14-AES256
root@host# set authentication-method pre-shared-keys
root@host# set authentication-algorithm sha-256
root@host# set dh-group group14
root@host# set encryption-algorithm aes-256-cbc
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to the Key Server. You should also configure the pre-shared key with a strong one. A weak one is used in this example. This should be the same as configured on the sub-servers and Key Server:

```
root@host# edit security group-vpn member ike policy KeySrv
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 1234567890
root@host# top
```

Now define how to reach the Key Server and sub-server. If you have multiple sub-servers, you normally divide the load over each of them, depending on the amount of GMs, but you can add up to the scale limit per sub-server. In a lab, we recommend that you add all GMs to one sub-server for easy verification and testing:

```
root@host# edit security group-vpn member ike gateway KeySrv
root@host# set ike-policy KeySrv
root@host# set server-address x.x.x.x (should match your Key or sub-server depending on
    topology, if multiple sub-servers, add the following
    in the order you want your redundancy).
```

```
root@host# set local-address 10.18.101.1
root@host# top
```

This defines the IPsec configuration:

```
root@host# edit security group-vpn member ipsec vpn GROUP_ID-0001
```

This defines which gateway to use:

```
root@host# set ike-gateway KeySrv
```

This defines which external interface to use, where encryption/decryption will take place:

```
root@host# set group-vpn-external-interface ge-0/0/1.0
```

Here you define which group you want to subscribe to when you have authenticated with the Key server:

```
root@host# set group 1
```

The recovery probe helps renegotiate a new IPsec key if you fall out of state:

```
root@host# set recovery-probe
```

If you have a need for excluding traffic from the encryption, you can use an exclude policy – this is just an example that could be used if there is any need, but you don't have to configure it:

```
root@host# set fail-open rule 1 source-address 2.2.2.0/24
root@host# set fail-open rule 1 destination-address 1.1.1.0/24

root@host# top
```

You should now add new security policies as you have deleted them above. Keep in mind that the `reject` option should be changed to `deny` in a live deployment:

```
root@host# set security policies default-policy deny-all

root@host# edit security policies global policy 1000
root@host# set match source-address any
root@host# set match destination-address any
root@host# set match application any
root@host# set match from-zone any
root@host# set match to-zone any
root@host# set then reject
root@host# set then log session-init
root@host# set then count
root@host# top
```

For GroupVPNs to even trigger its negotiation with the Key server on the SRX, you must have a steering IPsec-policy. If this one is not configured, but you have a security policy allowing traffic to and from the WAN, the default fail-close will want to trigger, so that's why this is really important:

```
root@host# edit security ipsec-policy from-zone LAN to-zone WAN
root@host# set ipsec-group-vpn GROUP_ID-0001
root@host# top
```

You also have to configure a security policy to define what should be allowed in and out between these zones. As the IPsec policy will encrypt/decrypt anything to/from 172.16.0.0/12; you can either define this network or make it wider or more narrow. It's all up to the security you have. Here we will use this networks as the control:

```
root@host# set security address-book global address 172.16.0.0/12 172.16.0.0/12

root@host# edit security policies from-zone LAN to-zone WAN policy 1
root@host# set match source-address 172.16.0.0/12
root@host# set match destination-address 172.16.0.0/12
root@host# set match application any
root@host# set then permit
root@host# set then log session-init
root@host# set then log session-close
root@host# top

root@host# edit security policies from-zone WAN to-zone LAN policy 1
root@host# set match source-address 172.16.0.0/12
root@host# set match destination-address 172.16.0.0/12
root@host# set match application any
root@host# set then permit
root@host# set then log session-init
root@host# set then log session-close
root@host# top
```

Finally you must configure the security zones to host the interfaces again, since we deleted the security stanza above. Here we allow IKE for the WAN-facing interface and PING and SSH as well for both interfaces for troubleshooting. This is something you might want to change for security purposes in a live deployment:

```
root@host# edit security zones security-zone LAN
root@host# set host-inbound-traffic system-services ssh
root@host# set host-inbound-traffic system-services ping
root@host# set interfaces ge-0/0/0.0
root@host# top

root@host# edit security zones security-zone WAN
root@host# set host-inbound-traffic system-services ike
root@host# set host-inbound-traffic system-services ssh
root@host# set host-inbound-traffic system-services ping
root@host# set interfaces ge-0/0/1.0
root@host# top
```

Commit the config:

```
root@host# commit
```

Configuring the GMs, Next One is MX

First delete all current configurations under the security stanza:

```
root@host# delete security
```

Change the hostname to make it easier to troubleshoot (not mandatory):

```
root@host# set system host-name GM-0003
```

Set system clock to current state before enabling NTP:

```
root@host# run set date (YYYYMMDDhhmm.ss)
```

If reachable, configure an NTP server, as Certificate authentication needs correct time to work:

```
root@host# set boot-server x.x.x.x
root@host# set server x.x.x.x
root@host# set server x.x.x.x
```

Now configure an IKE proposal to negotiate IKE:

```
root@host# edit security group-vpn member ike proposal PSK-SHA256-DH14-AES256
root@host# set authentication-method pre-shared-keys
root@host# set authentication-algorithm sha-256
root@host# set dh-group group14
root@host# set encryption-algorithm aes-256-cbc
root@host# top
```

You now have to bind the previous IKE proposal into an IKE policy that you can bind to the Key server. You should also configure the pre-shared key with a strong one. A weak one is used in this example. This should be the same as configured on the sub-servers and Key server:

```
root@host# edit security group-vpn member ike policy KeySrv
root@host# set mode main
root@host# set proposals PSK-SHA256-DH14-AES256
root@host# set pre-shared-key ascii-text 1234567890
root@host# top
```

Now define how to reach the Key Server and sub-server. If you have multiple sub-servers, you normally divide the load over each of them depending on the amount of GMs, but you can add up to the scale limit per sub-server. In a lab, we recommend that you add all GMs to one sub-server for easy verification and testing:

```
root@host# edit security group-vpn member ike gateway KeySrv
root@host# set ike-policy KeySrv
root@host# set server-address x.x.x.x (should match your Key or Subserver depending on topology, if
multiple sub-servers, add the following once is the order you want your redundancy).
root@host# set local-address 10.18.103.1
root@host# top
```

This defines the IPsec configuration:

```
root@host# edit security group-vpn member ipsec vpn GROUP_ID-0001
```

This defines which gateway to use:

```
root@host# set ike-gateway KeySrv
```

This helps the system verify the direction of traffic:

```
root@host# set match-direction output
```

Here you define which group you want to subscribe to when you have authenticated with the Key server:

```
root@host# set group 1
```

To prevent fragmentation you should configure this once for the MX:

```
root@host# set tunnel-mtu 1400
root@host# set df-bit clear
```

If you have a need for excluding traffic from the encryption, you can use a exclude policy – this is just an example that could be used if there is any need, but you don't have to configure it:

```
root@host# set fail-open rule 1 source-address 2.2.2.0/24
root@host# set fail-open rule 1 destination-address 1.1.1.0/24
root@host# top
```

As MXs do not use the FLOW process as the SRX does, you don't need to configure any firewall polices.

Here we need a `service-filter` to steer traffic to the service card. First you bypass the IKE traffic between the KS and GM, then you define what traffic should be steered to the service card from encryption:

```
root@host# edit firewall family inet service-filter GroupVPN-KS
root@host# set term inbound-ks from source-address 10.1x.10x.1/32 (this is equal to KS or SubServers)
root@host# set term inbound-ks then skip
root@host# set term outbound-ks from destination-address 10.1x.10x.1/32 (this is equal to KS or
SubServers)
root@host# set term outbound-ks then skip
root@host# set term GROUP_ID-0001 from source-address 172.16.0.0/12
root@host# set term GROUP_ID-0001 from destination-address 172.16.0.0/12
root@host# set term GROUP_ID-0001 then service
root@host# top
```

Now you must enable the service card, keeping in mind that you might be in a different slot, so verify this first:

```
root@host# set interfaces ms-0/2/0 unit 0 family inet
```

Configure a `service-set` ,which is the same as the information you configured under IPsec above:

```
root@host# set services service-set GROUP_ID-0001 interface-service service-interface ms-0/2/0.0
root@host# set services service-set GROUP_ID-0001 ipsec-group-vpn GROUP_ID-0001
```

Finally, bind this `service-set` and `service-filter` to the WAN interface:

```
root@host# edit interfaces xe-0/0/1 unit 0 family inet
root@host# set service input service-set GROUP_ID-0001 service-filter GroupVPN-KS
root@host# set service output service-set GROUP_ID-0001 service-filter GroupVPN-KS
root@host# top
```

Time to commit the configuration:

```
root@host# commit
```

Verification

Let's verify that your configuration is working. If something is not working, go to the troubleshooting section that follows. We will do verification of a stand alone key server as well as for a server cluster key server. Then we will do both the SRX and MX.

Verify Key Server in Server Cluster and Stand Alone

To make this section shorter, we will explain the difference between server cluster and stand alone by commands and syntaxes.

The `server-cluster` command shows communication between root server and sub-servers when they have connected, which also displays what is updated or not between devices. We have only shown one sub-server below. If this command is executed on another sub-server instead, you will only be able to see the information to the root server and not to any other sub-server. If a sub-server is not connected to the root server, it won't accept any registrations from a group member:

```
root@RootSrv# run show security group-vpn server server-cluster detail
Group: GROUP_ID-0001, Group Id: 1
Role: Root-server, Version Number: 120


Peer gateway: SubSrv01
 Peer IP: 10.16.101.1, Local IP: 10.10.101.1, VR: default
 Role: Sub-server, Status: Active
 CLUSTER-INIT send      : 0
 CLUSTER-INIT recv      : 2
 CLUSTER-INIT success     : 2
 CLUSTER-INIT fail      : 0
 CLUSTER-INIT dup     : 0
 CLUSTER-INIT abort     : 0
 CLUSTER-INIT timeout     : 0
 CLUSTER-UPDATE send      : 119
 CLUSTER-UPDATE recv      : 0
 CLUSTER-UPDATE success   : 119
 CLUSTER-UPDATE fail      : 0
 CLUSTER-UPDATE abort     : 0
 CLUSTER-UPDATE timeout   : 0
 CLUSTER-UPDATE pending   : 0
 CLUSTER-UPDATE max retry reached  : 0
```

```
 DPD send         : 24999
 DPD send fail        : 0
 DPD ACK recv       : 24994
 DPD ACK invalid seqno     : 0
 IPsec SA policy mismatch      : 0
 IPsec SA proposal mismatch    : 0
 KEK SA proposal mismatch    : 0
```

You normally only see IKE security associations for a short period of time until it's clear to the system to reduce resources between group members and it's a Key server or sub-server. Between the root server and sub-servers, you should always see an IKE SA.

Root-server:

```
root@RootSrv# run show security group-vpn server ike security-associations
Index State Initiator cookie   Responder cookie  Mode Remote Address
738955 UP 60b1fb7f7892bd5e  b635e6d72549671e Main 10.16.103.1
738954 UP fb3c210dab77f9f5  b75e8a532f1e031b Main 10.16.101.1
738953 UP 56ac0a9aa643f0c6  33fcfd3f0a88450f Main 10.16.102.1
738951 UP d93f7c01666cefed  999124751affc668 Main 10.16.104.1
```

Sub-server or stand alone Key server:

```
root@KeyServer# run show security group-vpn server ike security-associations detail
```

A `KEK security association` should always be up between root server > sub-server > group member or from keyserver > group member as long as server member communication is configured, or else you won't see any KEK security association:

```
root@KeyServer# run show security group-vpn server kek security-associations detail
Index 739317, Group Name: GROUP_ID-0001, Group Id: 1
Initiator cookie: 174e102b1bc66e95, Responder cookie: 916a9836dbb16fcc
Authentication method: RSA
Lifetime: Expires in 4194 seconds, Activated
Rekey in 3324 seconds
 Algorithms:
 Sig-hash    : sha256
 Encryption    : aes256-cbc
 Traffic statistics:
 Input bytes      : 168
 Output bytes       : 536
 Input packets      : 2
 Output packets     : 2
 Server Member Communication : Unicast
 Retransmission Period: 10,  Number of Retransmissions: 2
 Group Key Push sequence number: 2

PUSH negotiations in progress  : 0
```

Next we should verify that the IPsec security-associations are distributed between all devices. In the command itself, you have to define if it's used on the server or member side:

```
root@KeyServer# run show security group-vpn server ipsec security-associations detail
Group: GROUP_ID-0001, Group Id: 1
```

```
Total IPsec SAs: 1
 IPsec SA: GROUP_ID-0001
 Protocol: ESP, Authentication: sha256, Encryption: aes-256
 Anti-replay: D3P enabled, window size 1000 milliseconds
 SPI: 8e521ccc
 Lifetime: Expires in 1038 seconds, Activated
 Rekey in 528 seconds
 Policy Name: 1
 Source: 172.16.0.0/12
 Destination: 172.16.0.0/12
 Source Port: 0
 Destination Port: 0
 Protocol: 0
```

Verify that all Group Members are registered successfully. This could be done either on the sub-server or the stand alone Key server.

```
root@KeyServer# run show security group-vpn server registered-members detail
Group: GROUP_ID-0001, Group Id: 1
 Total number of registered members: 2

 Member gateway: GM-0001, Member IP: 10.18.101.1, Vsys: root
 Last Update: Thu Jan 07 2016 05:45:54
 Stats:
 Pull Succeeded  : 3
 Pull Failed   : 0
 Push Sent   : 68
 Push Acknowledged  : 68
 Push Unacknowledged  : 0
```

Here we get statistics over all the GMs from a PULL/PUSH perspective. This could be done either on the sub-server or the stand alone Key server:

```
root@KeyServer# run show security group-vpn server statistics
Group: GROUP_ID-0001, Group Id: 1
 Stats:
 Pull Succeeded   : 14
 Pull Failed      : 315
 Pull Exceed Member Threshold  : 0
 Push Sent      : 314
 Push Acknowledged    : 313
 Push Unacknowledged  : 1
```

On the Group Members

Here you can verify traffic statistics:

```
root@GM-0001# run show security group-vpn member ipsec statistics
ESP Statistics:
 Encrypted bytes:   20640
 Decrypted bytes:   6132
 Encrypted packets:  120
 Decrypted packets:   73
AH Statistics:
 Input bytes:     0
 Output bytes:    0
```

```
 Input packets:   0
 Output packets:  0
Errors:
 AH authentication failures:  0
 ESP authentication failures:  0
 ESP decryption failures:  0
 Bad headers: 0, Bad trailers: 0
D3P Statistics:
 Old timestamp packets:   0
 New timestamp packets:  0
 No timestamp packets:    0
 Unexpected D3P header packets:  0
 Invalid type packets:    0
 Invalid length packets:    0
 Invalid next header packets:  0
Exclude Statistics:
 Created sessions:    0
 Invalidated sessions:    0
Dynamic Policy Statistics:
 Created sessions:    120
 Invalidated sessions:    0
Fail-Open Statistics:
 Created sessions:    0
 Invalidated sessions:    0
Fail-Close Statistics:
 Dropped packets:   0
```

Troubleshooting

Here are some typical issues and mistakes to check out.

Group Member

First check that you get a IPsec SA and evaluate that it's correct:

```
root@GM-0001# run show security group-vpn member policy
Group VPN Name: GROUP_ID-0001, Group Id: 1
From-zone: LAN, To-zone: WAN
 Tunnel-id: 49152, Policy type: Secure
 Source : IP <172.16.0.0 - 172.31.255.255>, Port <0 - 65535>, Protocol <0>
 Destination : IP <172.16.0.0 - 172.31.255.255>, Port <0 - 65535>, Protocol <0>

 Tunnel-id: 63489, Policy type: Fail-open (Inactivated)
 Source : IP <2.2.2.0 - 2.2.2.255>, Port <0 - 65535>, Protocol <0>
 Destination : IP <1.1.1.0 - 1.1.1.255>, Port <0 - 65535>, Protocol <0>

 Tunnel-id: 63488, Policy type: Fail-close
 Source : IP <0.0.0.0 - 255.255.255.255>, Port <0 - 65535>, Protocol <0>
 Destination : IP <0.0.0.0 - 255.255.255.255>, Port <0 - 65535>, Protocol <0>
```

If your policy seems to be correct, check that you don't have a problem with anti-replay. In that case, the numbers below should change. Then you most likely need to increase your anti-replay timer if your group member is on a virtual platform. Also check that you have the correct timing configured:

```
root@GM-0001# run show security group-vpn member ipsec statistics
D3P Statistics:
 Old timestamp packets:   0
 New timestamp packets:  0
 No timestamp packets:    0
 Unexpected D3P header packets:  0
 Invalid type packets:    0
 Invalid length packets:    0
 Invalid next header packets:  0
```

If you don't see any IPsec SA, or if it does not match what you expect, verify your group ID for that SA and then check your Key server side.

If you see an IPsec SA on the group member, skip this step, or else verify that your group member is connected:

```
root@KeyServer# run show security group-vpn server registered-members
Group: GROUP_ID-0001, Group Id: 1
 Total number of registered members: 2
 Member Gateway   Member IP Last Update  Vsys
 GM-0001    10.18.101.1 Thu Jan 07 2016 08:55:50 root
 GM-0003    10.18.103.1 Thu Jan 07 2016 08:55:49 root
```

If your group member is not connected, go to the Key server steps below "Stand Alone Key Server" or "Sub-server."

If it's connected, then it might be that you missed the `ike-gateway` for your GM or that it's incorrectly configured:

```
root@KeyServer# show security group-vpn server group GROUP_ID-0001
group-id 1;
member-threshold 2000;
ike-gateway GM-0001;
ike-gateway GM-0003;
ike-gateway GM-0005;
anti-replay-time-window 8000;
server-member-communication {
 communication-type unicast;
 lifetime-seconds 7200;
 encryption-algorithm aes-256-cbc;
 sig-hash-algorithm sha-256;
}
ipsec-sa GROUP_ID-0001 {
 proposal AES256-SHA256-L3600;
 match-policy 1 {
 source 172.16.0.0/12;
 destination 172.16.0.0/12;
 protocol 0;
 }
}
```

Server Cluster

For the server cluster you should first verify that you have an established IKE session between the root server and each sub-server. This should always be up. If it goes down from time to time, verify your network connectivity. If it does not come up from the start, verify that you have the correct paramemters, such as:

- Correct IKE proposal
- An IKE policy with the correct pre-shared key
- All of your sub-servers under the IKE gateway parameters

Then, verify your group configuration. Be sure to verify that the `server-role` is correct on each host as well:

```
group GROUP_ID-0001 {
 group-id 1;
 member-threshold 2000;
 server-cluster {
  server-role root-server;
  ike-gateway SubSrv01;
  ike-gateway SubSrv02;
  ike-gateway SubSrv03;
  ike-gateway SubSrv04;
  retransmission-period 10;
 }
 anti-replay-time-window 8000;
 server-member-communication {
  communication-type unicast;
  lifetime-seconds 1000;
  encryption-algorithm aes-256-cbc;
  sig-hash-algorithm sha-256;
```

Check that you allow incoming IKE messages on the security zone. Then check any network connectivity issues.

Stand Alone Key Server or Sub-server

Here is an error you'll most likely see when your firewall filter is incorrect on your GM:

```
KeyServer gksd 50002 - - IKE negotiation failed with error: Invalid syntax. IKE Version: 1, VPN:
Not-Available Gateway: Not-Available, Local: 10.10.105.1/848, Remote: 10.18.104.1/848, Local IKE-ID:
Not-Available, Remote IKE-ID: Not-Available, VR-ID: 0: Role: Responder
```

Verify that your sub-servers have connectivity with the root server, or else they won't accept any group member registrations.

Then check if each group-id hasn't reached the total amount of group members according to the member-threshold under each group, as that will also reject no registrations.

Verify that you have the correct paramemters such as:

- Correct IKE proposal
- An IKE policy with the correct pre-shared key
- All your sub-servers under the IKE gateway parameters

Chapter 5

Deploy Remote Access VPN

Remote Access VPN is used for both an organization's administrators and its users, and each of the connections must be able to traverse network elements that block IKE/IPsec. The NCP client has a fallback option to encapsulate these packets into a TCP packet over port 443, meaning that most network elements on hotspots and hotels will allow these packets.

This final step guides you to the appropriate deployments based on what you prefer. It covers both the SRX and NCP management server as well as the client, but will not discuss the distribution of the client software. For some configurations, you'll need to reference *Step 2: Configure PKI, RADIUS, and EAP-TLS*, that describes how to set up and enable external authentication.

There are several combinations that can be used, as shown here:

	Local IP-Pool (SRX)	External IP-Pool (via RADIUS)	User Authentication using Local Password (SRX)	User Authentication using RADIUS (Ex to Active Directory)	User Authentication using (Ex to Active Directory)	Pre-Shared Key	Certificates
IKEv1	Supported	Supported	Supported	Supported	N/A	Supported	N/A
IKEv2	Supported	Supported	N/A	Not part of this book.	Supported	N/A	Supported

Remote Access Topology Using NCP

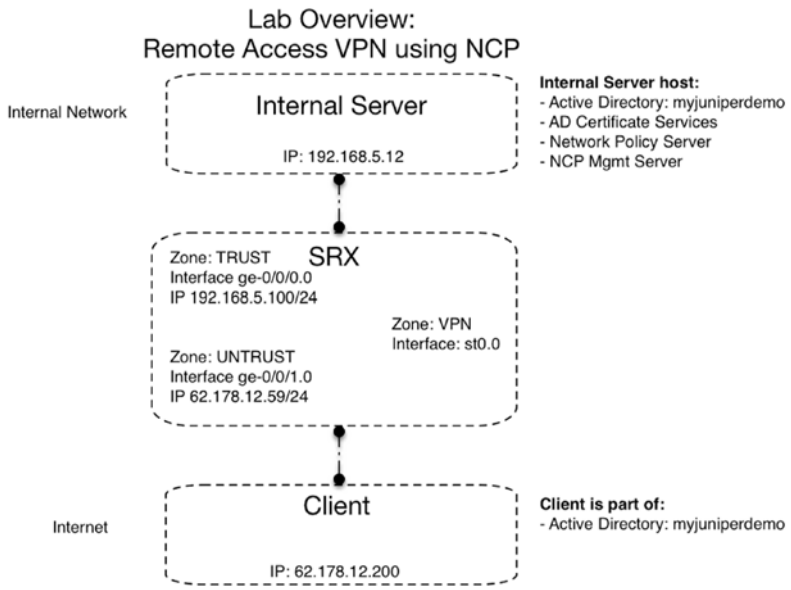


Figure 5.1 *Remote Access VPN Using NCP*

Requirements

Hardware: Juniper SRX Service Gateway.

Software: Junos 15.1X49D80 and above

NCP Management: Server

NCP Client: NCP Exclusive Remote Access Client

Radius Server: Needed for Active Directory authentication os users

Certificate Authority: Needed for IKEv2 EAP-TLS

SCEP (Simple Certificate Enrollment Protocol) and OCSP (Online Certificate Status Protocol) are used for signing and revocation verification. Manual signing and certificate revocation lists are also possible to use, but will not be described in this book. Keep also in mind that this book has the CA accessible through the 11.10.0.0/24 network. If you have your CA behind any other router or firewall, make sure that it is accessible.

NOTE Only hardware or their virtual versions that support the above referenced software are supported for this solution.

Remote Access VPN – Base criteria

If you will be using External Authentication or IP-Pools follow the Microsoft examples in this book.

Decide if You Want to Use Local or External IP-Pools

For Local IP-Pool configure the following.

Define an IP-Pool:

```
user@host# top edit access address-assignment pool RA_LOCAL-IP-POOL family inet
```

Create a prefix that should be used for the remote access users:

```
user@host# set network 10.17.106.0/24
```

Decide which IPs in this network should be usable by the clients:

```
user@host# set range REMOTEACCESS low 10.17.106.10
user@host# set range REMOTEACCESS high 10.17.106.254
```

Define which DNS and WINS server is to be used for the client when they connect:

```
user@host# set xauth-attributes primary-dns 192.168.5.12
user@host# set xauth-attributes secondary-dns x.x.x.x (only for IKEv2)
user@host# set xauth-attributes primary-wins x.x.x.x
user@host# set xauth-attributes secondary-wins x.x.x.x (only for IKEv2)
user@host# top
```

If External IP-Pool:

Then the configuration is done on the external side, no need to make any statements here.

Decide if You Want Local, RADIUS, or EAP-TLS User Authentication

(Local authentication is not supported with IKEv2.)

For Local Authentication

Create a profile for local authentication. You will also be given an option to decide if you want to use a local or external IP-POOL here.. If you use an external IP-Pool, you might want to chage the profile name to reflect that (we will not specify it in any of these examples). We will only define the profile based on local vs external authentication:

```
user@host# top edit access profile RA_LOCAL-AUTH
user@host# set client "username" firewall-user password << "Password for the user"
<<(add more users if needed)
```

Next (optional) add the local IP-POOL you want to use. This is the one we created above:

```
user@host# set address-assignment pool RA_LOCAL-IP-POOL
user@host# top
```

For RADIUS or EAP-TLS Authentication

Create a profile for your RADIUS server that can authenticate your users. You will be using the Microsoft RADIUS Server. See "Network Policy Server" on how to install and set this up:

```
user@host# top edit access profile RA_EXTERNAL-AUTH
```

Now define that you will use RADIUS as the authtentication protocol:

```
user@host# set authentication-order radius
```

Now define the critera for the RADIUS server request:

```
user@host# edit radius-server 192.168.5.12
user@host# set secret "Use 22 characters that start and end with an alpha character"
<<(This should be the same secret on the radius server side)

user@host# set port xxxx
<<(If your server listen to a non-default port)
```

Next (optional) add the local IP-POOL you want to use. This is the one we created above:

```
user@host# up 2
user@host# set address-assignment pool RA_LOCAL-IP-POOL
user@host# top
```

Bind the external interface and ST0 interfaces to zones and define the IKE and TCP-Encap profiles to be used to allow incoming connections with the fallback to TCP-Encap.

Create your Internal and External and ST0 interfaces and bind them respectively to TRUST, UNTRUST, and VPN zones as in the topology explained at the start of this section.

Now enable the TCP-Encap profile to be used for fallback:

```
user@host# top set security tcp-encap profile NCP log
```

This allows Junos to listen to IKE and TCP-Encap for the incoming connections:

```
user@host# top edit security zone security-zone UNTRUST host-inbound-traffic
user@host# set system-service ike
user@host# set system-service tcp-encap
user@host# top
```

Now you need to create your firewall policy to allow traffic between each zone and network that your users should have access to, and determine if the trusted network should be able to reach the clients. Keep in mind that you need to have routing pointing to client prefixes behind the SRX used for the remote access.

Remote Access VPN – Using IKEv1 with Local User Authentication

This is probably the best way to do a basic test of the functionality, but not convenient for any real deployment because the users will be forced to use a local username and password on the SRX that they can't change on their own, meaning helpdesk and some other function needs access to the SRX. So this should only be seen as a demo or test deployment for the general user.

NOTE Configuration is displayed in snippets below. Keep in mind that you need to change variable data such as interface, IP address and other information related to your own network if you do not follow this book.

Let's make the rest of the necessary SRX configuration to be able to setup/establish the remote access connection. We start by defining the security IKE proposals, policy, and IKE gateway definitions. This needs to match on the client side further down as well.

Create the IKE Proposal:

```
user@host# top edit security ike proposal PSK-DH14-AES256-SHA256-L28800
user@host# set authentication-method pre-shared-keys
user@host# set dh-group group14
user@host# set authentication-algorithm sha-256
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 28800
```

Create the IKE Policy:

```
user@host# up 1 edit policy RA_IKEv1_LOCAL-AUTH
user@host# set mode aggressive
user@host# set proposals PSK-DH14-AES256-SHA256-L28800
user@host# set pre-shared-key ascii-text <<"ThisShouldBeAStrongPassword"
```

Create the IKE Gateway:

```
user@host# up 1 edit gateway RA_IKEv1_LOCAL-AUTH
user@host# set external-interface ge-0/0/1.0
user@host# set version v1-only
user@host# set dynamic ike-user-type shared-ike-id
user@host# set dynamic user-at-hostname remoteaccess@ra.myjuniperdemo.net << (This should be euniq for
your organization)
user@host# set ike-policy RA_IKEv1_LOCAL-AUTH
user@host# set tcp-encap-profile NCP <<(profile created earlier)
user@host# set aaa access-profile RA_LOCAL-AUTH <<(profile created earlier)
user@host# top
```

Now let's proceed with the IPsec part of the configuration:

```
user@host# edit security ipsec proposal ESP-AES256-SHA256-L3600
user@host# set protocol esp
user@host# set authentication-algorithm hmac-sha-256-128
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 3600
```

Create IPsec Policy:

```
user@host# up 1 edit policy RemoteAccess
user@host# set perfect-forward-secrecy key group14
user@host# set proposals ESP-AES256-SHA256-L3600
```

Create the IPsec VPN object:

```
user@host# up 1 edit vpn RA_IKEv1_LOCAL-AUTH
user@host# set bind-interface st0.1
user@host# set ike gateway RA_IKEv1_LOCAL-AUTH
user@host# set ike ipsec-policy RemoteAccess
user@host# set traffic-selector NO-SPLIT local-ip 0.0.0.0/0 <<(If you specify a shorter prefix, the
client will do split tunnel of his outgoing traffic)
user@host# set traffic-selector NO-SPLIT remote-ip 0.0.0.0/0 <<(This must be 0.0.0.0/0)
user@host# top
user@host# commit
```

Now you have completed the SRX side of the configuration, please go to the section "Set up the NCP Client" below that reference on how to configure the NCP client.

Remote Access VPN – Using IKEv1 with RADIUS User Authentication

This is probably be the most common way to deploy Remote Access because this requires a minimum amount of configuration and provides ease of use for the users. Keep in mind that IKEv1 has its security challenges, in this case the user device authenticates with the SRX with a pre-shared key using aggresive mode for IKEv1, and that can be exposed. Still, to gain access to resources behind the SRX, the user needs to authenticate to the Active Directory, and a strong policy for password and log on attempts will of course reduce the security challenge.

NOTE Configuration is displayed in snippets below. Keep in mind that you need to change variable data such as interface, IP address and other information related to your own network if you do not follow this book.

NOTE For the users to authenticate with Active Directory, you need to complete the Network and Policy Server setup in Chapter 2.

Let's make the rest of the necessary SRX configuration to be able to set up and establish the remote access connection.

You start by defining the security IKE proposals, policy, and IKE gateway definitions. This needs to match on the client side further down as well.

Create the IKE Proposal:

```
user@host# top edit security ike proposal PSK-DH14-AES256-SHA256-L28800
user@host# set authentication-method pre-shared-keys
user@host# set dh-group group14
user@host# set authentication-algorithm sha-256
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 28800
```

Create the IKE Policy:

```
user@host# up 1 edit policy RA_IKEv1_EXTERNAL-AUTH
user@host# set mode aggressive
user@host# set proposals PSK-DH14-AES256-SHA256-L28800
user@host# set pre-shared-key ascii-text <<"ThisShouldBeAStrongPassword"
```

Create the IKE Gateway:

```
user@host# up 1 edit gateway RA_IKEv1_EXTERNAL-AUTH
user@host# set external-interface ge-0/0/1.0
user@host# set version v1-only
user@host# set dynamic ike-user-type shared-ike-id
user@host# set dynamic user-at-hostname "remoteaccess@ra.myjuniperdemo.net"
user@host# set ike-policy RA_IKEv1_EXTERNAL-AUTH
user@host# set tcp-encap-profile NCP <<(profile created earlier)
user@host# set aaa access-profile RA_EXTERNAL-AUTH <<(profile created earlier)
user@host# top
```

Now proceed with the IPsec part of the configuration:

```
user@host# edit security ipsec proposal ESP-AES256-SHA256-L3600
user@host# set protocol esp
user@host# set authentication-algorithm hmac-sha-256-128
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 3600
```

Create the IPsec Policy:

```
user@host# up 1 edit policy RemoteAccess
user@host# set perfect-forward-secrecy key group14
user@host# set proposals ESP-AES256-SHA256-L3600
```

Create the IPsec VPN object:

```
user@host# up 1 edit vpn RA_IKEv1_EXTERNAL-AUTH
user@host# set bind-interface st0.1
user@host# set ike gateway RA_IKEv1_EXTERNAL-AUTH
user@host# set ike ipsec-policy RemoteAccess
user@host# set traffic-selector NO-SPLIT local-ip 0.0.0.0/0 << (If you specify a shorter prefix, the
client will do split tunnel of his outgoing traffic)
user@host# set traffic-selector NO-SPLIT remote-ip 0.0.0.0/0 << (This must be 0.0.0.0/0)
user@host# top
user@host# commit
```

Now you have completed the SRX side of the configuration.

Next go to the section "Setup the NCP Client" to reference how to configure the NCP client.

Remote Access VPN – Using IKEv2 with EAP-TLS User Authentication

If you're using certificate based authentication, you should chose to deploy this example. By doing this, you will use IKEv2 using EAP-TLS where each user authenticates itself using a personal certificate instead of a traditional password in the active directory.

NOTE Configuration is displayed in snippets below. Keep in mind that you need to change variable data such as interface, IP address, and other information related to your own network if you do not follow this book.

NOTE For the users to authenticate with Active Directory, you need to complete the Network and Policy Server setup in Chapter 2.

Let's make the rest of the necessary SRX configuration to be able to set up and establish the remote access connection.

We start by defining the configuration for certificates to allow the client to authenticate the SRX, then configure the security IKE proposals, policy, and IKE gateway definitions. This needs to match on the client side further down as well.

Let's configure and request the certificates that we need on the SRX and we need to define how to find the CA:

```
user@host# edit security pki ca-profile MyJuniperDemo-CA_Server
user@host# set ca-identity MyJuniperDemo-CA_Server
```

Specify the CA SCEP URL:

```
user@host# set enrollment url http://192.168.5.12/certsrv/mscep/mscep.dll
```

Configure how to verify the validity of the certificate:

```
user@host# set revocation-check use-ocsp ocsp
user@host# set revocation-check ocsp url http://192.168.5.12/ocsp
```

Remember that the challenge-password is unique to your CA:

```
user@host# top edit security pki auto-re-enrollment scep certificate-id RemoteAccessNCP
user@host# set ca-profile-name MyJuniperDemo-CA_Server
user@host# set re-generate-keypair
user@host# set re-enroll-trigger-time-percentage 10
user@host# set challenge-password 8CDB49EEEC84401A85D5F58800DB2F96
user@host# set scep-encryption-algorithm des3
user@host# top
user@host# commit
```

Before proceeding, we need to download the root certificate of the CA server:

```
user@host# run request security pki ca-certificate enroll ca-profile MyJuniperDemo-CA_Server
user@host# run request security pki ca-certificate verify ca-profile MyJuniperDemo-CA_Server

user@host# run show security pki statistics
```

If "`ocsp_resp_success: 0`" is zero, then you have some problem with your OCSP instance on the CA server, and if you do not want to troubleshoot this now, deactive OCSP verification for now:

```
(set security pki ca-profile MyJuniperDemo-CA_Server revocation-check disable)
```

This should result in "`CA certificate MyJuniperDemo-CA_Server verified successfully`" if the SRX has the certificate correctly. Keep in mind, this is *not* what should be used in production.

Now let's request a local certificate for the SRX that the client can verify later on. Copy and paste may not work for you:

```
user@host# run request security pki local-certificate enroll ca-profile MyJuniperDemo-CA_Server
certificate-id RemoteAccessNCP domain-name remoteaccessncp.myjuniperdemo.net ip-address 62.178.12.59
subject C=US,ST=California,L=Sunnyvale,O="Juniper Networks Inc",OU=MyJuniperDemo,CN=RemoteAccessNCP
challenge-password 37BAE506396A1DB7F0144A2793419FA9
```

Create the IKE Proposal:

```
user@host# top edit security ike proposal CERT-DH14-AES256-SHA256-L28800
user@host# set authentication-method rsa-signatures
```

```
user@host# set dh-group group14
user@host# set authentication-algorithm sha-256
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 28800
```

Create the IKE Policy:

```
user@host# up 1 edit policy RA_IKEv2_EXT-AUTH
user@host# set proposals CERT-DH14-AES256-SHA256-L28800
user@host# set certificate local-certificate RemoteAccessNCP
```

Create the Gateway:

```
user@host# up 1 edit gateway RA_IKEv2_EXT-AUTH
user@host# set external-interface ge-0/0/1.0
user@host# set version v2-only
user@host# set dynamic ike-user-type group-ike-id
user@host# set dynamic user-at-hostname remoteaccess@ra.myjuniperdemo.net <<(This should be euniq for
your organization)
user@host# set ike-policy RA_IKEv2_EXT-AUTH
user@host# set tcp-encap-profile NCP << (profile created earlier)
user@host# set aaa access-profile RA_EXTERNAL-AUTH <<(profile created earlier)
user@host# top
```

Now to the IPsec part of the configuration:

```
user@host# edit security ipsec proposal ESP-AES256-SHA256-L3600
user@host# set protocol esp
user@host# set authentication-algorithm hmac-sha-256-128
user@host# set encryption-algorithm aes-256-cbc
user@host# set lifetime-seconds 3600
```

Create the IPsec Policy:

```
user@host# up 1 edit policy RemoteAccess
user@host# set perfect-forward-secrecy key group14
user@host# set proposals ESP-AES256-SHA256-L3600
```

Create the IPsec VPN object:

```
user@host# up 1 edit vpn RA_IKEv2_EXT-AUTH
user@host# set bind-interface st0.1
user@host# set ike gateway RA_IKEv2_EXT-AUTH
user@host# set ike ipsec-policy RemoteAccess
user@host# set traffic-selector NO-SPLIT local-ip 0.0.0.0/0 << (If you specify a shorter prefix, the
client will do split tunnel of his outgoing traffic)
user@host# set traffic-selector NO-SPLIT remote-ip 0.0.0.0/0 <<(This must be 0.0.0.0/0)
user@host# top
user@host# commit
```

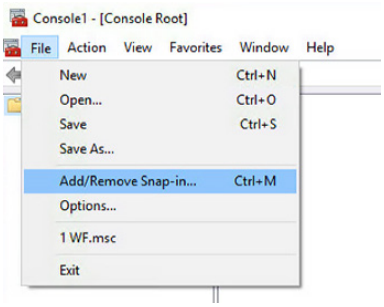
Now you have completed the SRX side of the configuration.

Next you can go to the section “Setup the NCP Client” to reference how to configure the NCP client.

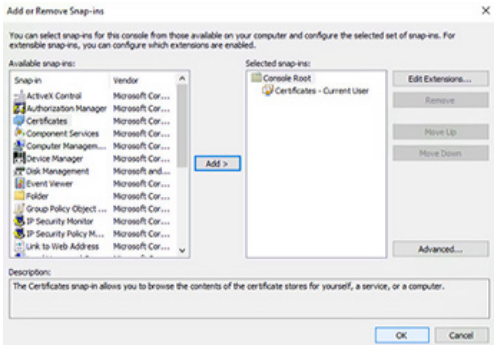
Prerequisite for Clients Using EAP-TLS on Windows

These next steps should be performed on your Windows client and will use IKEv2 EAP-TLS; they can be executed in multiple ways, but are done this way so as not to force any other installation of third-party software.

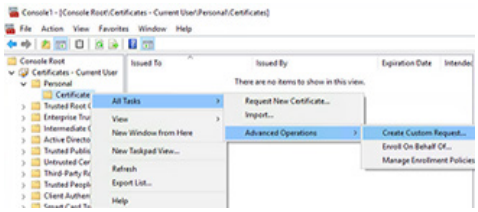
First, be sure your Windows client is part of your Active Directory. Then open an MMC console (this can be done by searcing for "mmc.")



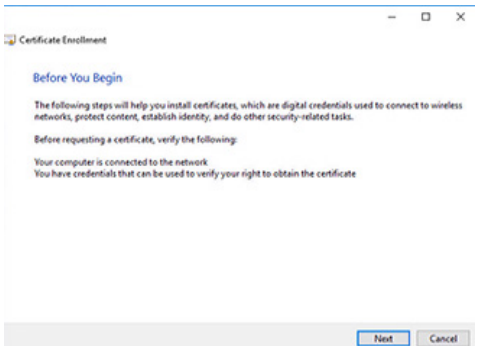
Click on the "File" menu then "Add/Remove Snap-in."



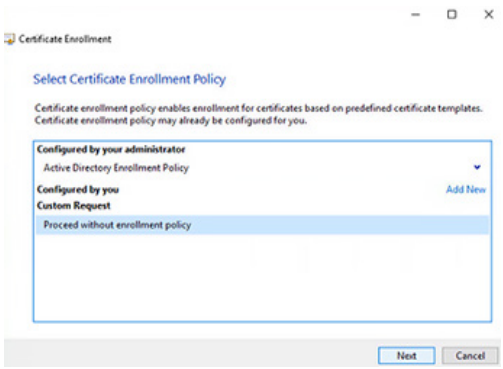
Highlight "Certificates" and click Add in the middle of the screen.



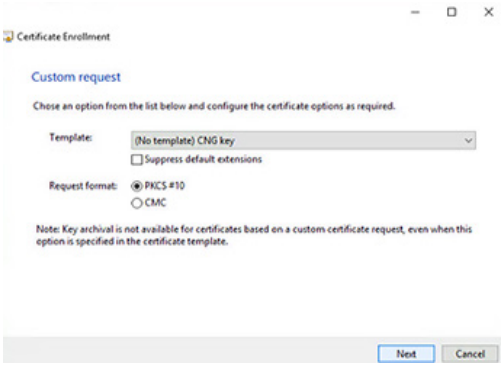
Highlight "Console Root" then right-click "Certificates" and then click 'Create Custom Request."



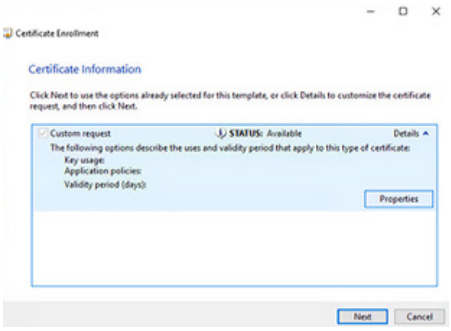
Click Next.



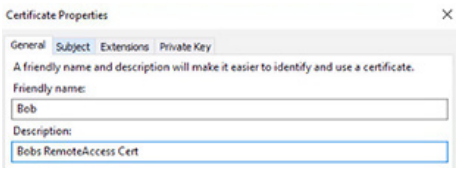
Highlight "Proceed without enrollment policy" and click Next.



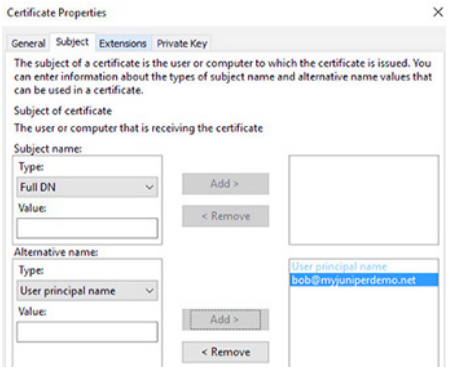
Click Next.



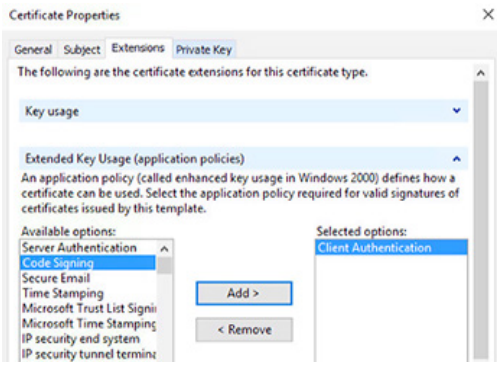
Click on "Details" to expand the information and then click "Properties."



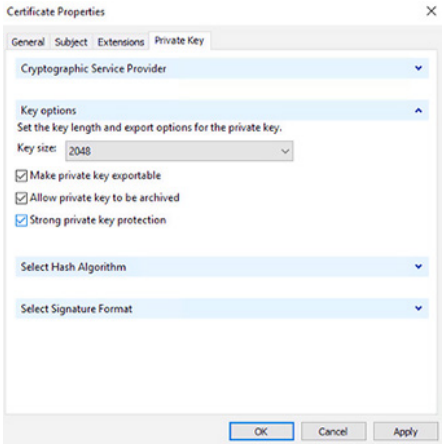
In the "Friendly name" field, give this certificate a logical name, we give it Bob as that's our user. Also give a name for the "Description" and then click on the "Subject" tab.



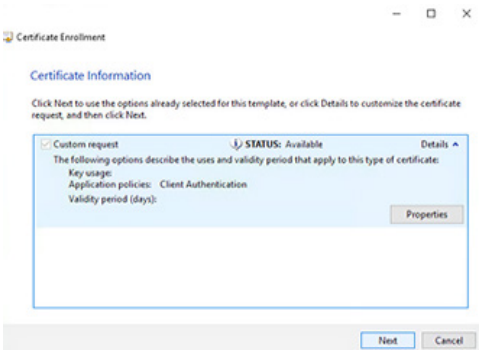
Under the "Alternative name" choose "User principal name" and enter the same user name as the user you have in the Active Directory, then click Add and click on the "Extensions" tab.



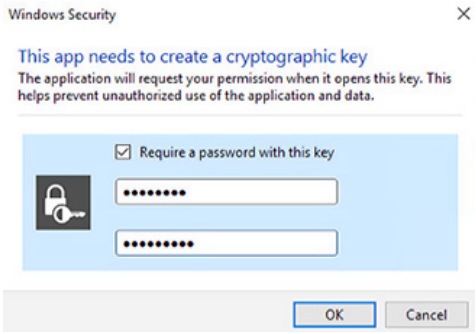
Expand the"Extended Key Usage (application policies)" information and then select "Client Authentication" and click Add. Now click on the "Private Key" tab.



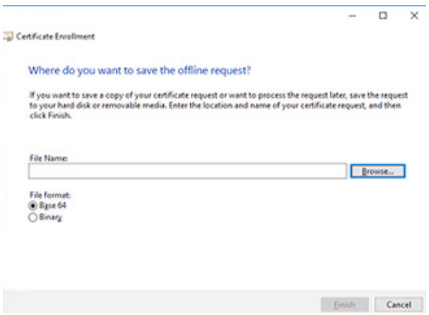
Expand the "Key options" information and choose "Key size" as "2048" then check the three options "Make Private key exportable," "Allow private key to be archived," and "Strong private key protection." Then click OK.



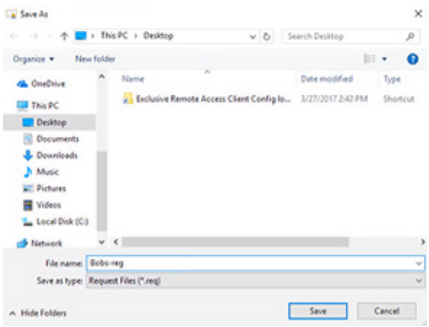
Click Next.



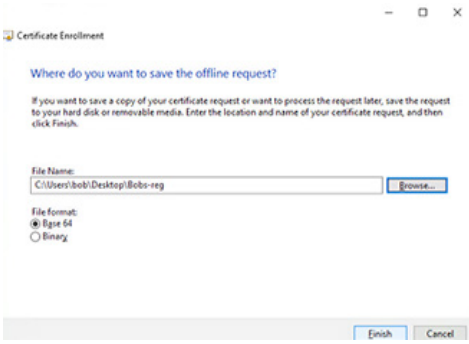
Now you need to set a password for the certificate that will be used to authenticate the use of this certificate later on. Choose one that you will remember. Click OK.



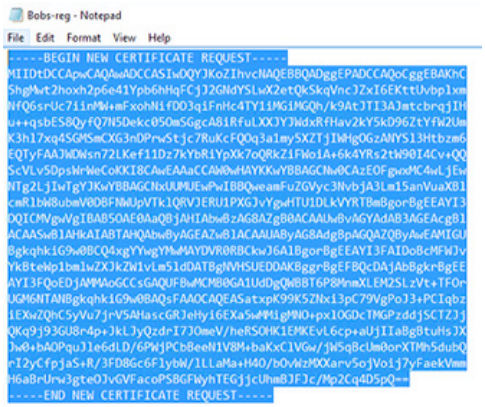
Click the "Browse" button to decide where you want to store the certificate request file.



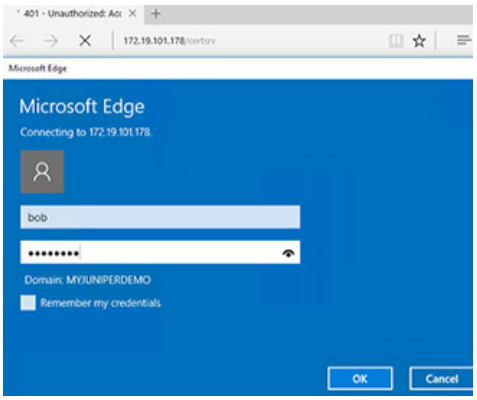
Choose a name that you'll remember for this user, then click Save.



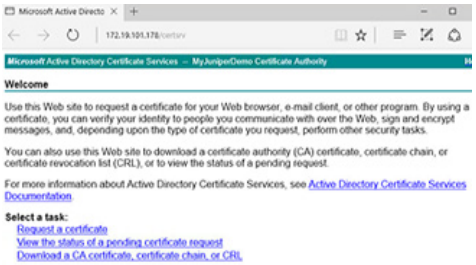
Click Finish and the file will be saved to the location you choose.



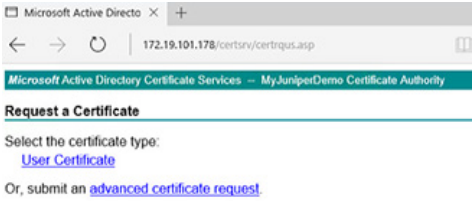
Now, go to that location and open the saved file with the Notepad. Highlight the text and “copy” to clipboard.



Open a browser and browse to http://fqdn-of-your-certificate-server/certsrv and log in with your user credentials.

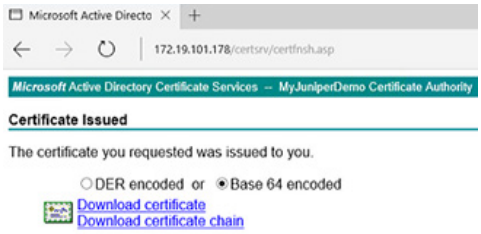


Click "Request a certificate."



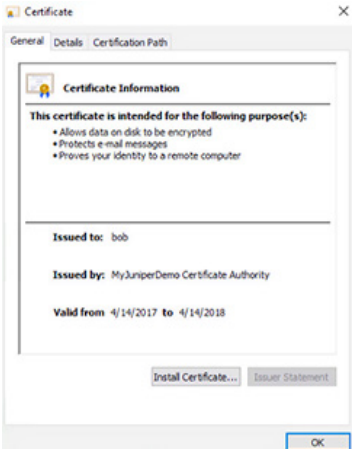
Click "advanced certificate request."

Now paste the information that you copied from the Notepad document into the "Saved Request" form and then click Submit.

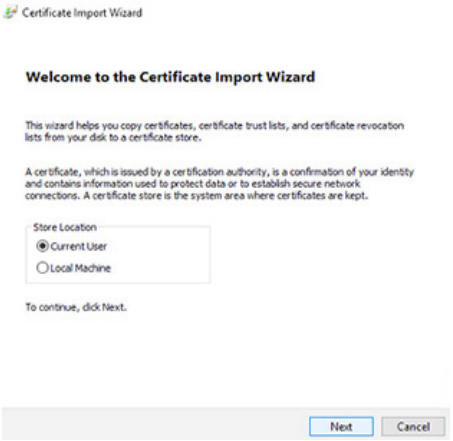


Check "Base 64 encode" and then click "Download certificate."

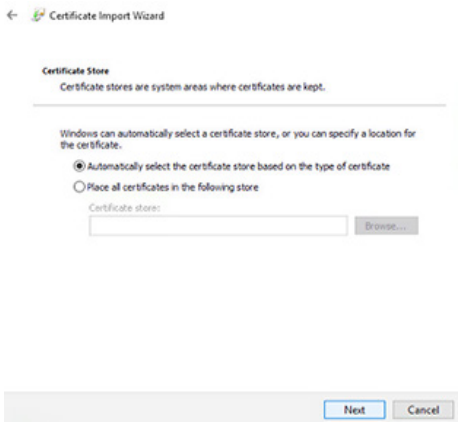
Locate the downloaded certificate and double-click on it to open.



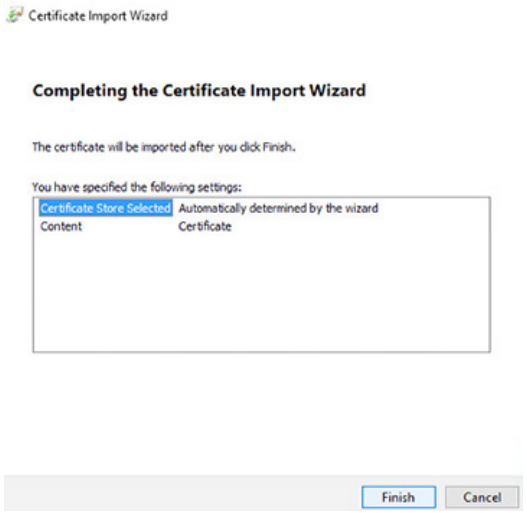
When open, click on "Install Certificate."



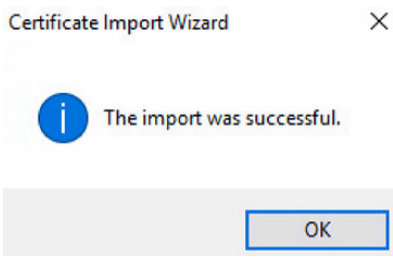
Click Next.



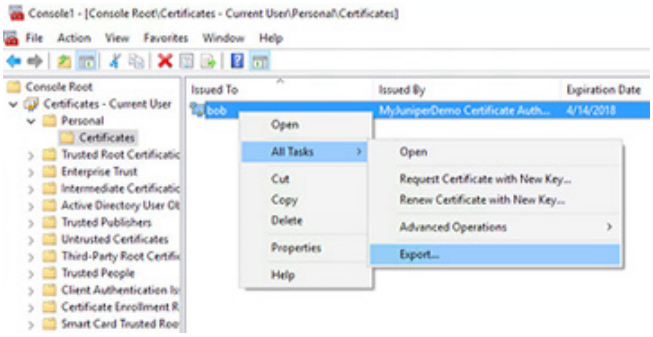
Click Next.



Click Finish.

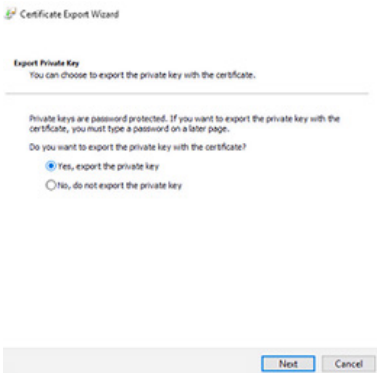


Click OK.

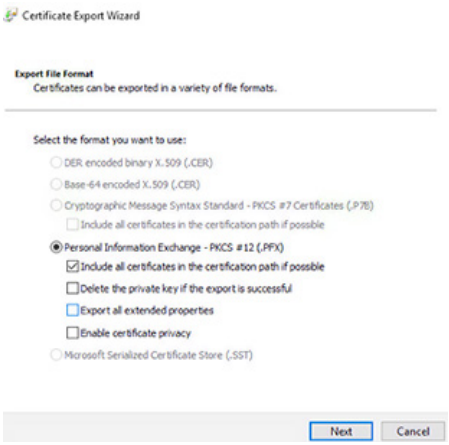


Now right-click on the imported certificate and choose “Export.”

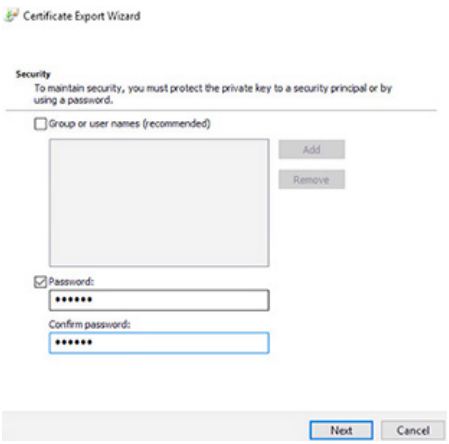
Click Next.



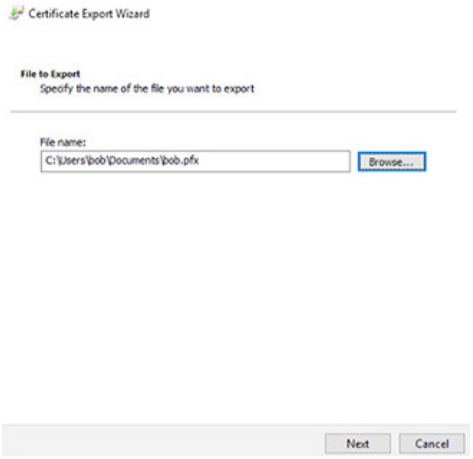
Check "Yes, export the private key" and then click Next.



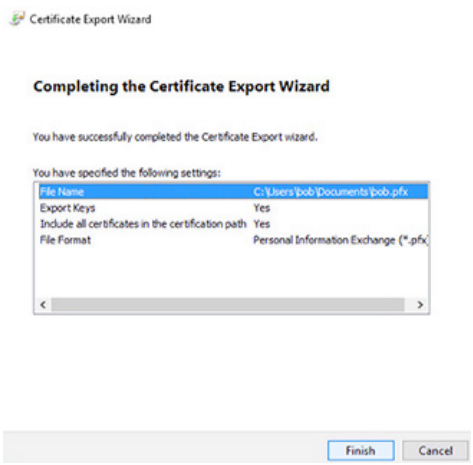
Click Next.



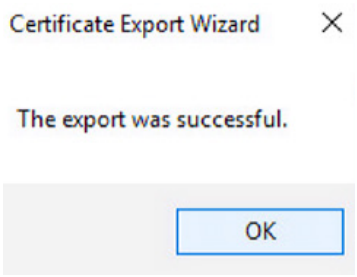
Check "Password" and then choose a password that will be used for this exported certificate.



Browse to choose a location and then click Next.



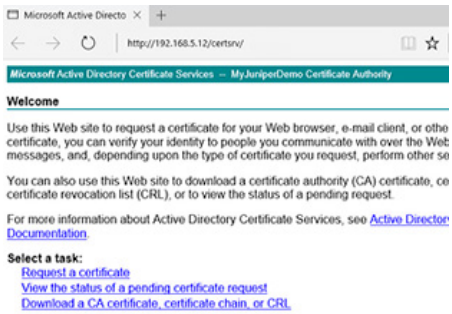
Click Finish.



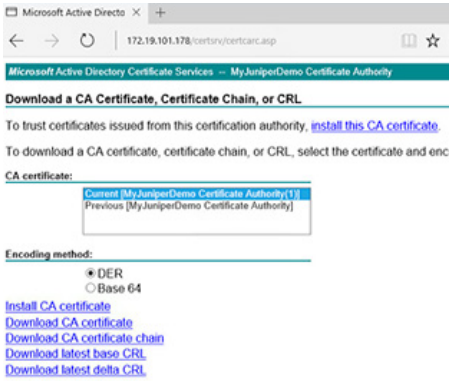
Click OK.

As IKEv2 requires certificates, you also need to download and install the Certificate Authorities' "ROOT Certificate."

Open a browser and browse to your CA server, in our case, http://192.168.5.12/CertSrv. If you need to authenticate yourself, do so, and the "Welcome" screen should appear.



Click on the "Download a CA certificate, certificate chain, or CRL" task.



Highlight the most current "CA certificate," check on "DER," and then click "Download CA certificate" from the list.

When the CA Certificate is downloaded, remember where it's saved because you will have to move it later.

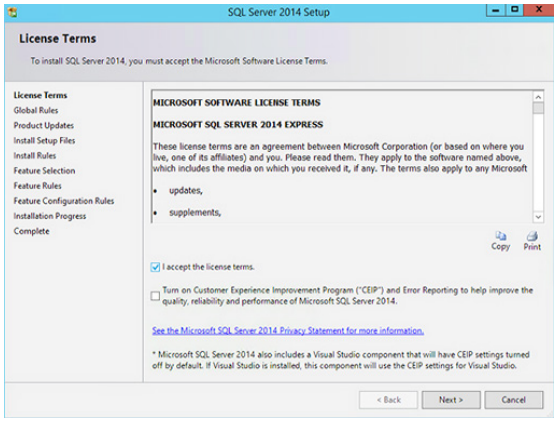
Installing NCP Management Server

You need to have the NCP management server up and running and accessible for all your clients when they are connected to the VPN for licensing and distribution. With this server you can also manage the connection profiles. You can do this, and you can do a lot more, but please see the NCP documentation for options, for creating more administrators, or for any more specific configurations.

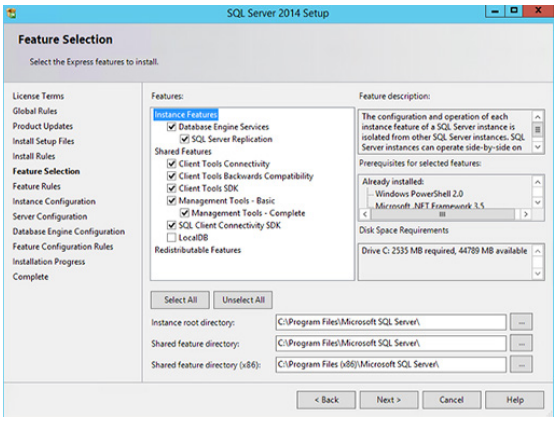
To install the management server, you first need to install the free Microsoft SQL Express 2014. You can find it with a basic web search. When you get to the download site, you will choose the installer: *ExpressAndTools 64BIT\SQLEXPRWT_x64_ENU.exe*.

When the file is downloaded, start the installation and go through all the default selections until you come to this window.

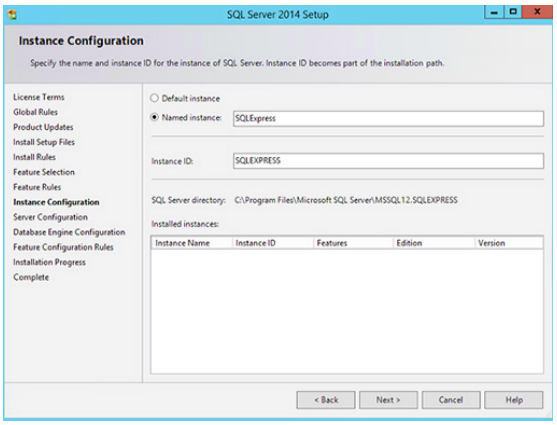
Here you choose *"New SQL Server stand-alone installation or add features to an existing installation."*



To proceed you must read and acknowledge the license terms. Check that you accept and then click Next.

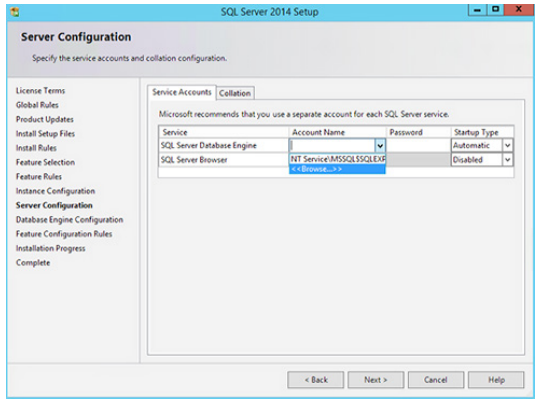


Click Next.

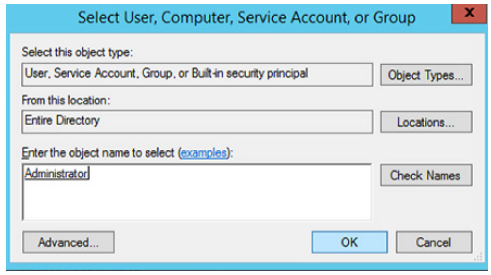


Here you can choose your own name for the new instance – we went with the default one. So just click Next.

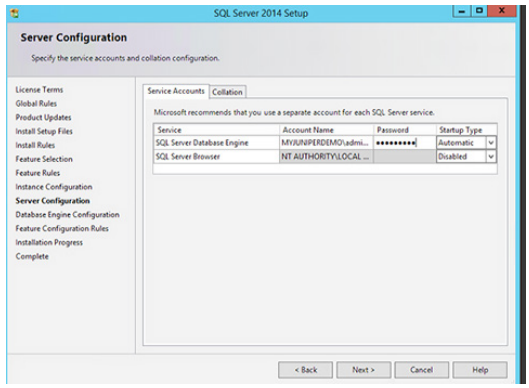
In the next step you choose an account that will own this DB. Normally you should create a "Service/Application" account, but for this lab, we'll use the 'administrators' account.



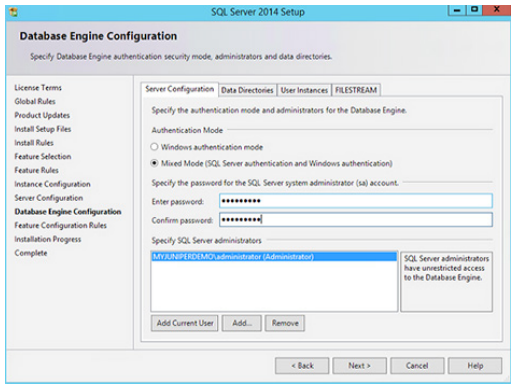
Click the "*Account Name*" and browse for the account.



Type the account name, and click "Check Names." This should find the account, then click OK.

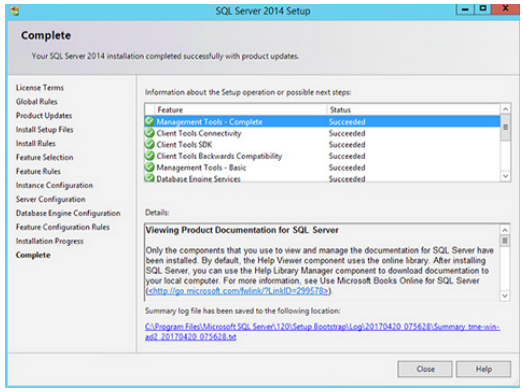


Type the password for the chosen account in the "*Password box*" and click Next.



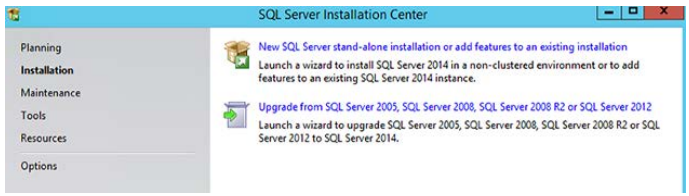
Check the mode “Mixed Mode (SQL Server Authentication and Windows authentication” and enter the password for the choosen account (added on the previous screen) and then click Next.

It may take some time for the installation to finish.



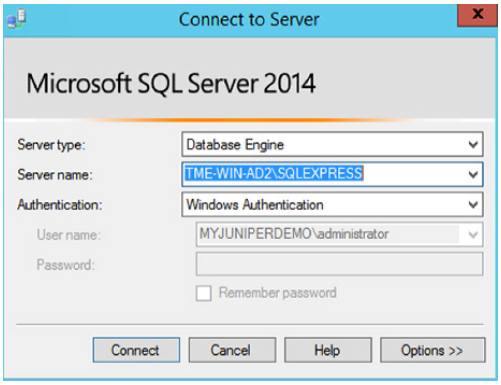
Click Close.

You should get the following screen.

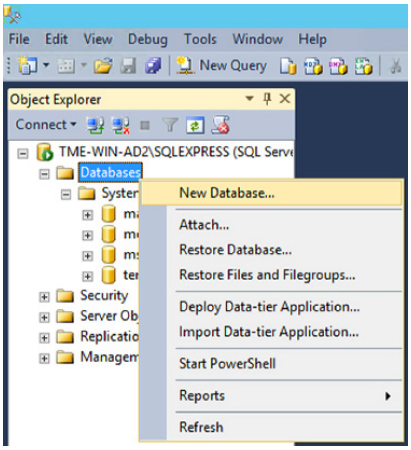


Close this window by clicking the upper right “X.”

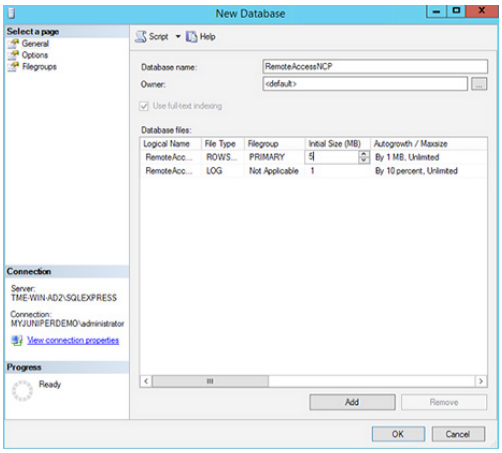
Now, launch the "Microsoft SQL Server Management Studio" application from the "Start" menu.



Click Connect.

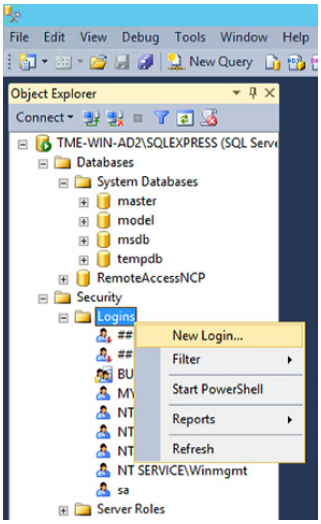


Right-click "Databases" and then choose "New Database."

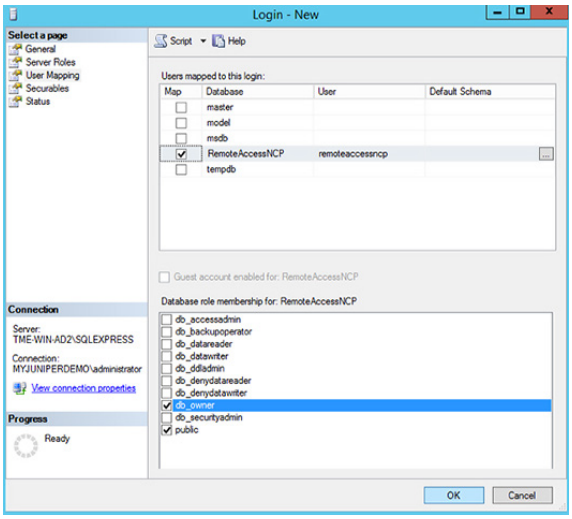


Give this new database a name, here we use "*RemoteAccessNCP.*"

In the "*Initial Size (MB)*" column of the database files, change this to 5, or else it will not work. Then click OK.



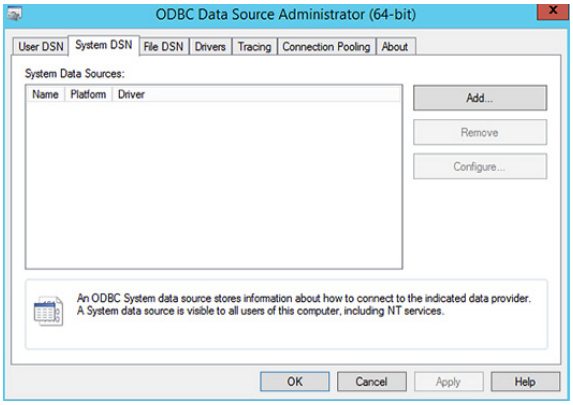
Choose "Security" > "Login" > "New Login."



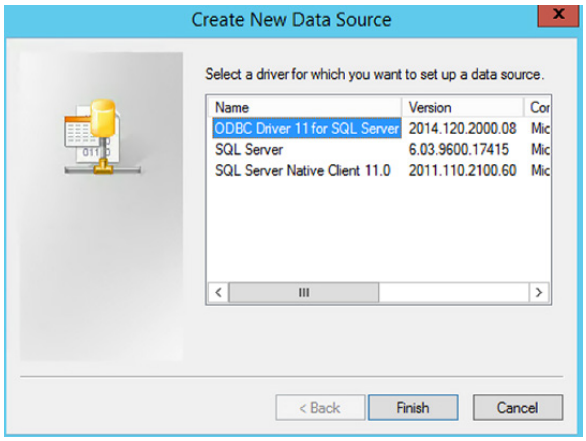
Select "*User Mapping*" in the left navigation pane. Under "Users mapped to this login" check the newly created database "*RemoteAccessNCP.*" Now check "*db_owner*" and click OK.

You can close the Microsoft SQL Server Management Studio application.

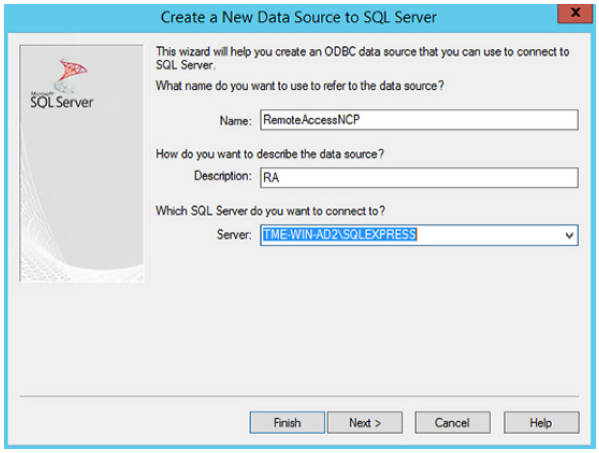
Now launch the "ODBC Data Sources (64-bit)" application.



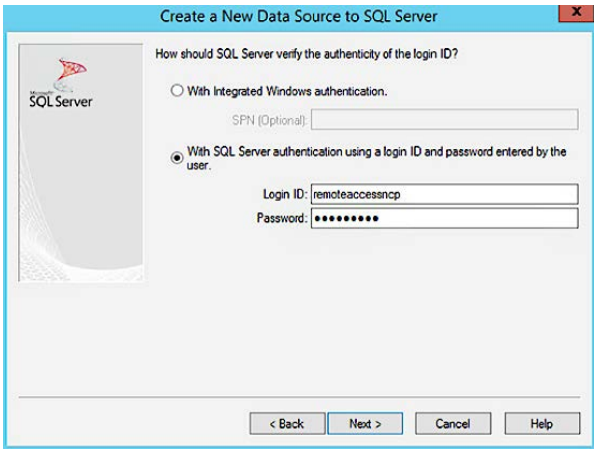
Navigate to the System DSN tab and click "Add."



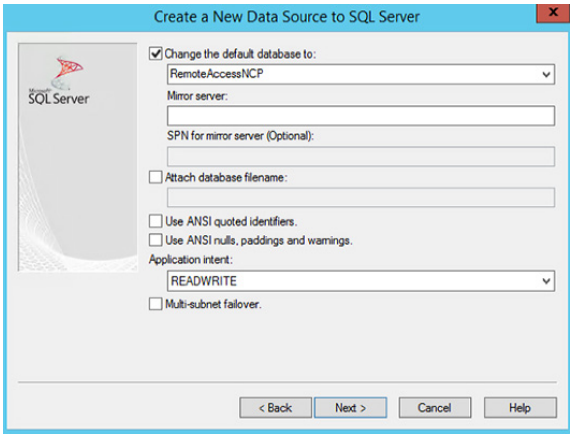
Choose "ODBC Driver 11 for SQL Server" and click Finish.



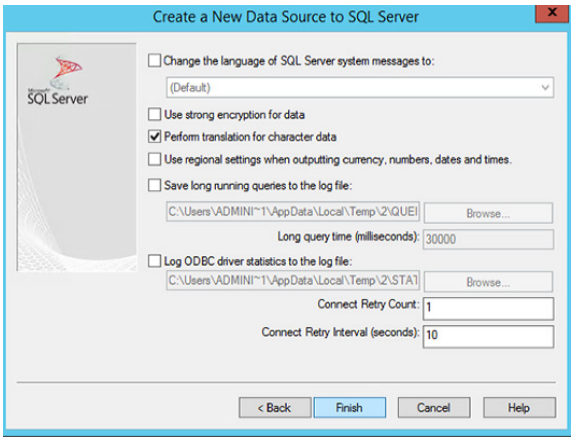
Give the ODBC connection a name. Here, we used "RemoteAccessNCP" with a description of "RA." For the "Server" field choose from the available list, you should only have one.



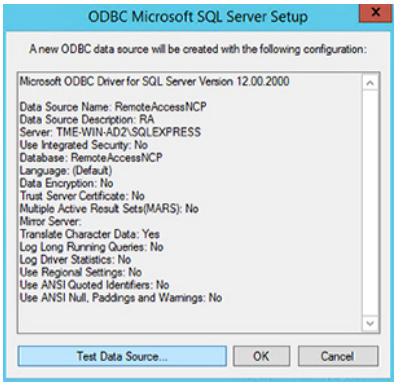
Check the "With SQL Server authentication using a login ID and password entered by the user" option. Here, we use the recently-added user with their password. Click Next.



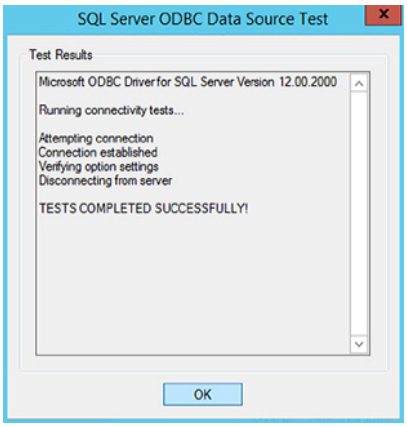
Change the default database to the newly created one "*RemoteAccessNCP.*" Also, *uncheck* "Use ANSI quoted identifiers" and "Use ANSI nulls, paddings and warnings" options, and then click Next.



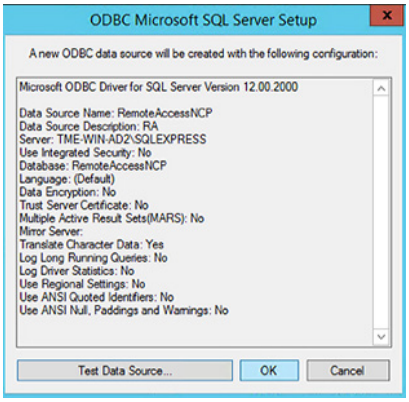
Click Finish.



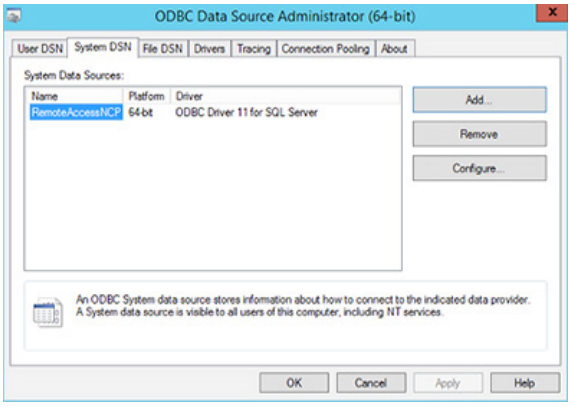
Now click on “Test Data Source.”



You should see "TESTS COMPLETED SUCCESSFULLY!" Click OK.

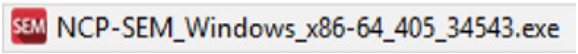


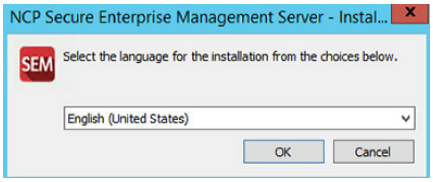
Click OK.



Click OK and exit the app.

Now let's start the installation of the NCP management server itself. Download and launch the installer (you might have a different version depending on when you are working through this book).

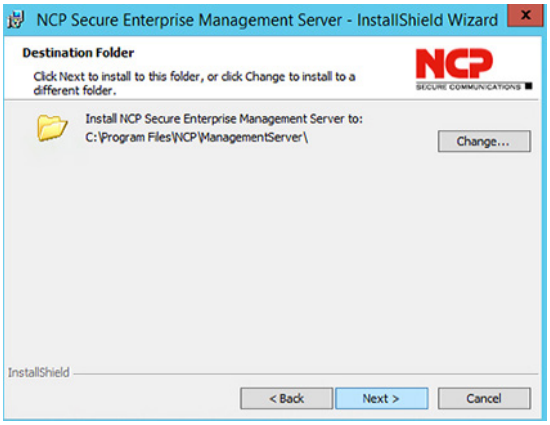




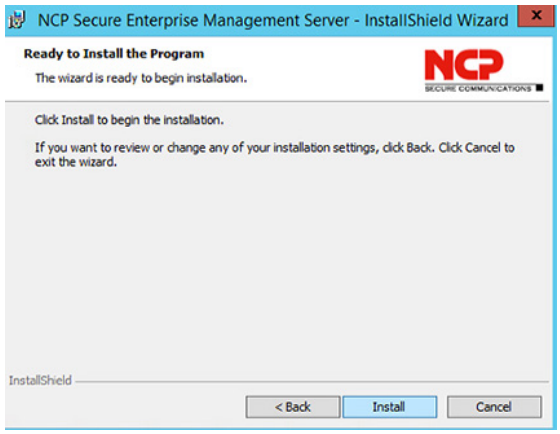
Click OK.

Click Next.

Read and accept the license terms, then click Next.

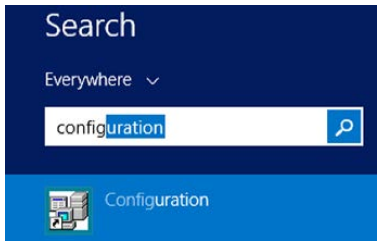


Click Next.

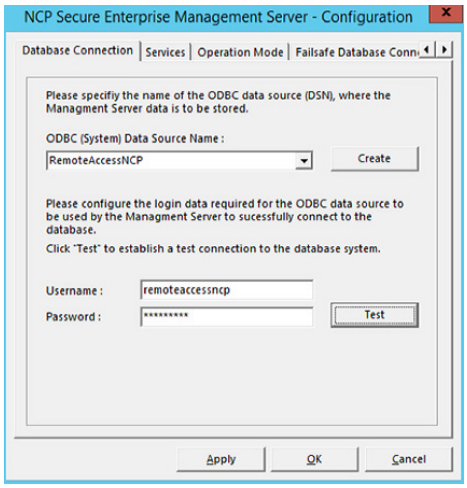


Click Install.

When the installation is complete, click Finish.

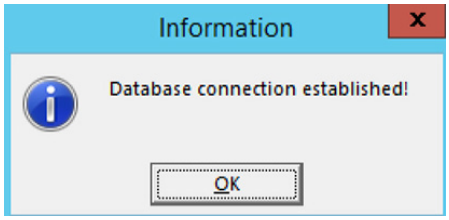


If the “NCP Secure Enterprise Management Server – Configuration” does not start by itself, search for “configuration” and launch the application.

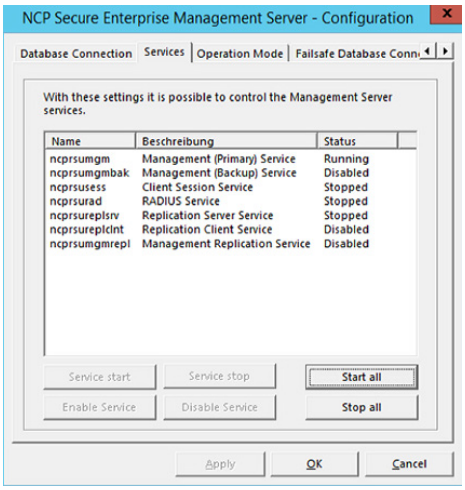


Choose your database in the "ODBC (System) Data Source Name" field and enter the username and password that you set before. Click "Test."

You should now see that you are able to establish the database connection.

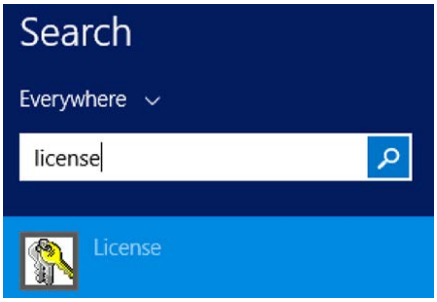


Now click OK and then click Apply in the previous screen.

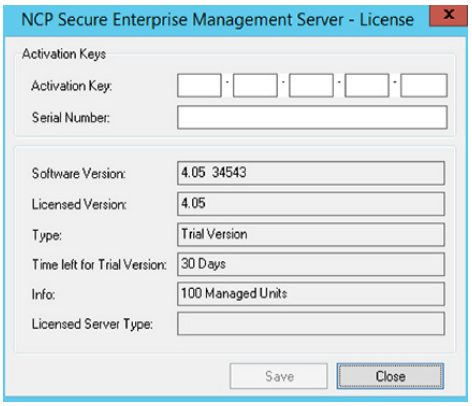


Navigate to the “Services” tab. Click “Start all.”

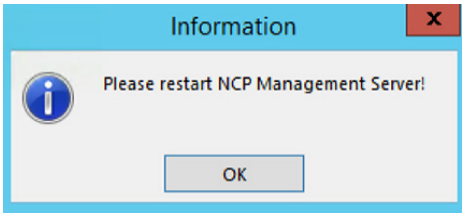
Keep that application open and search for the application “License.”



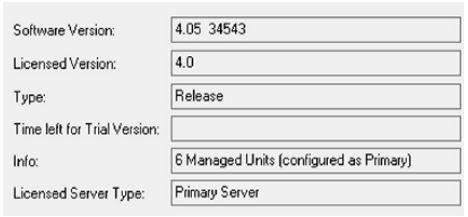
Launch the app.



Enter the license information that you got from NCP and click Save.

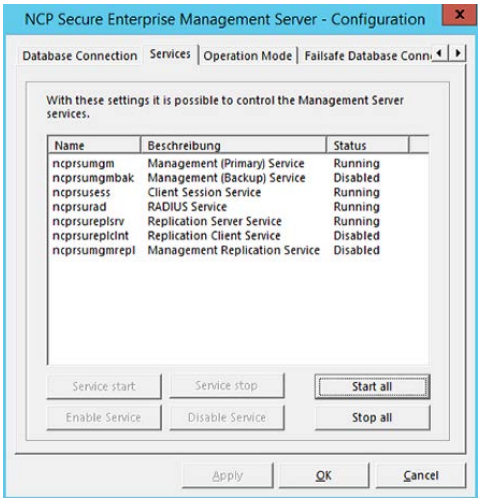


Click OK.



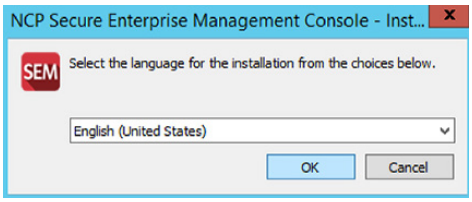
You should see that you have an active license as shown. Once you do, click "Close."

Now go back to the NCP configuration tool.



Click “Stop all” then “Start all,” and then finally, click OK.

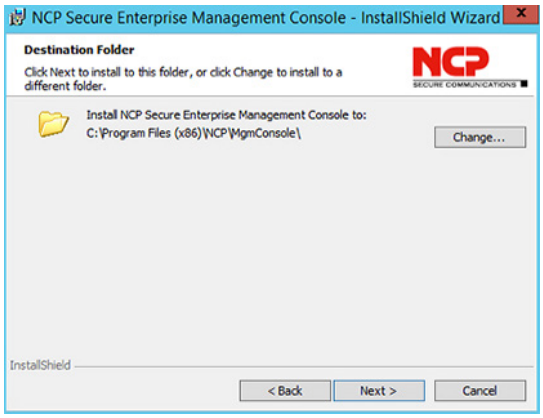
Now you need to launch the console application. You might have a different version than the one shown here.



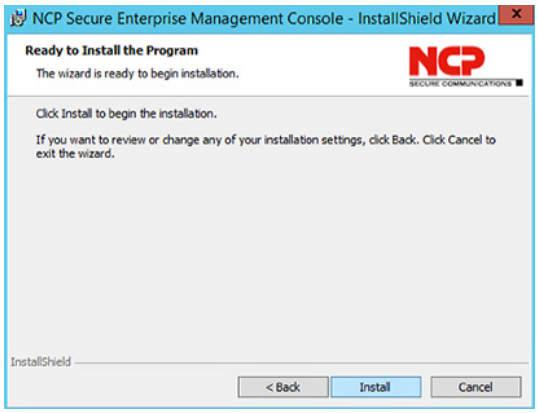
Click OK.

Click Next.

Read and accept the license terms, then click Next.



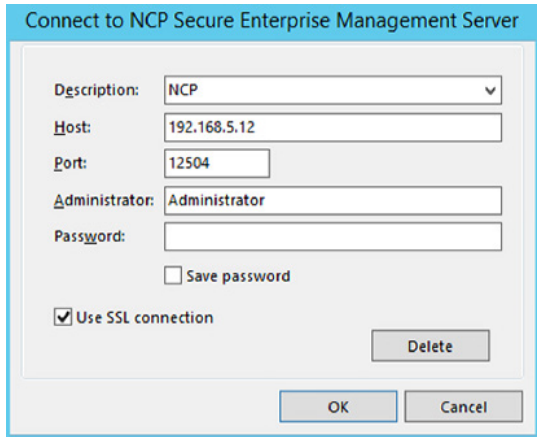
Click Next.



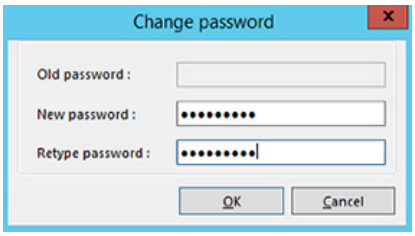
Click Install.

Click Finish.

Now launch the "NCP Secure Enterprise Management Console."



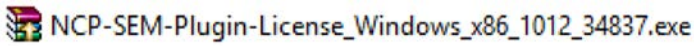
Give this connection a name. Here, we use "NCP." Specify the FQDN or IP address of the NCP Server in the "Host" field. (The first time you connect to the server, the administrator account is "Administrator" with a blank password so that's why we now click OK.)

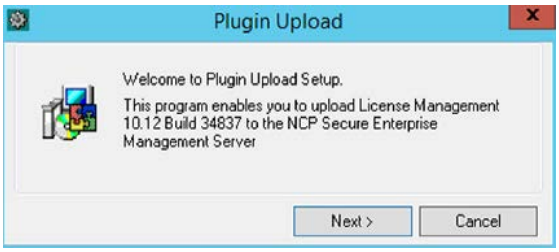


Set a password for the account "Administrator" and click "OK." This will log you in to the console. Close this console for now.

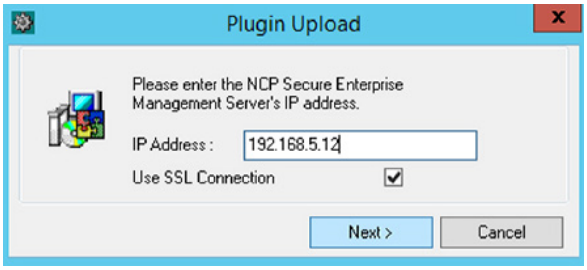
For the console to be useful, you must install a minimum of two plug-ins: "License" and "Client Configuration."

Let's start with the License plug-in. Launch the executable.

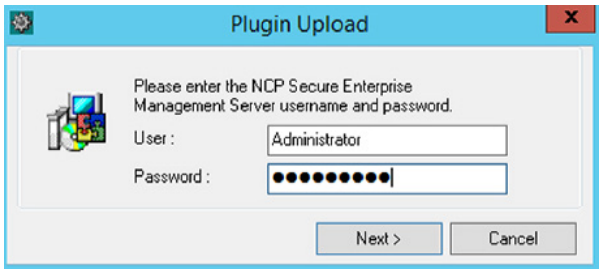




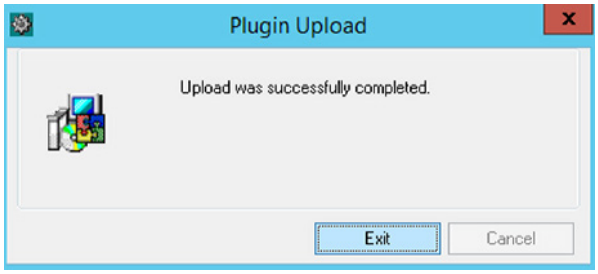
Click Next.



Type the IP Address for your NCP Secure Enterprise Management server and click Next.

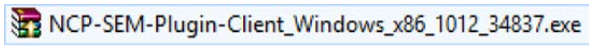


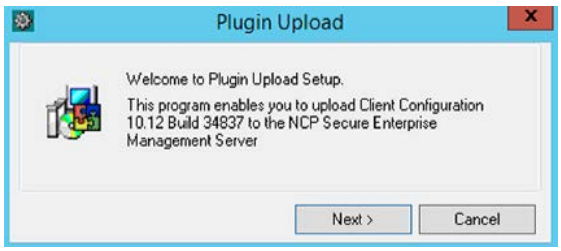
Enter the "User" and "Password" information and click Next.



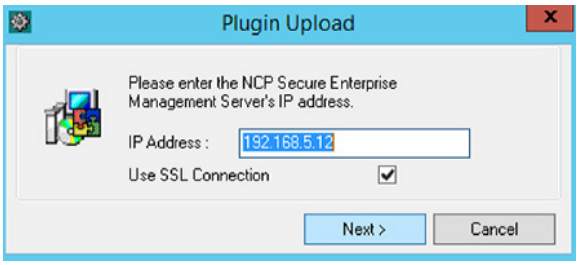
Click Exit.

Now install the NCP console application by running the installer.

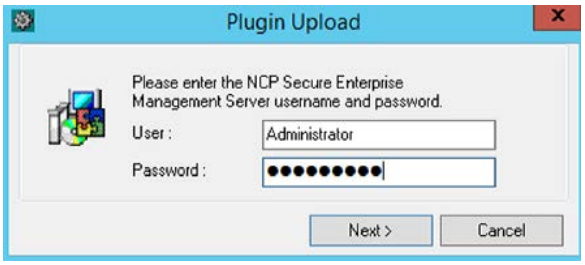




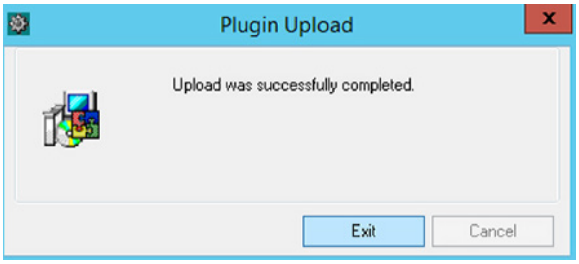
Click Next.



Enter the IP Address of the NCP Secure Enterprise Management Server and click Next.

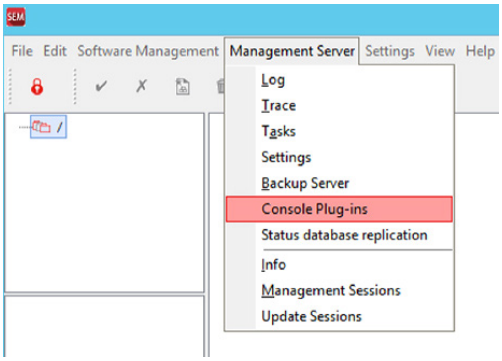


Enter the “User” and “Password” information and click Next.

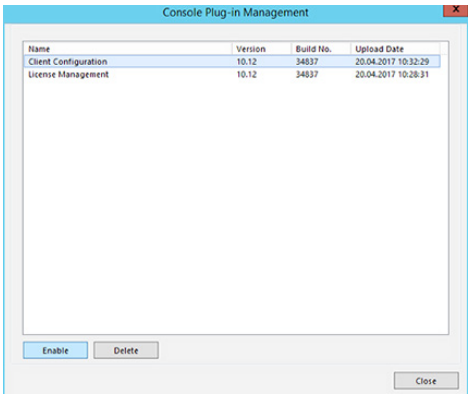


Click Exit.

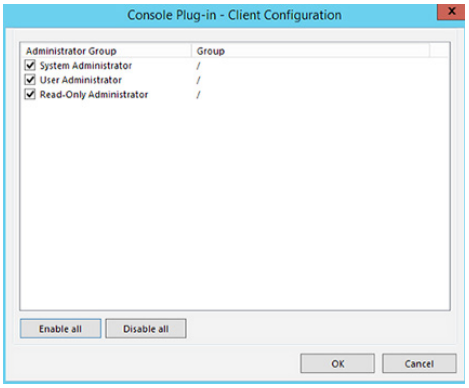
Start the NCP Console application and log in.



Select the "Management Server" tab and then "Console Plug-ins."



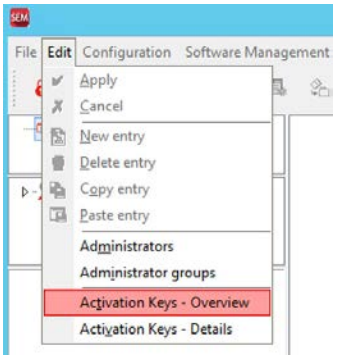
Highlight "Client Configuration" and click on "Enable."



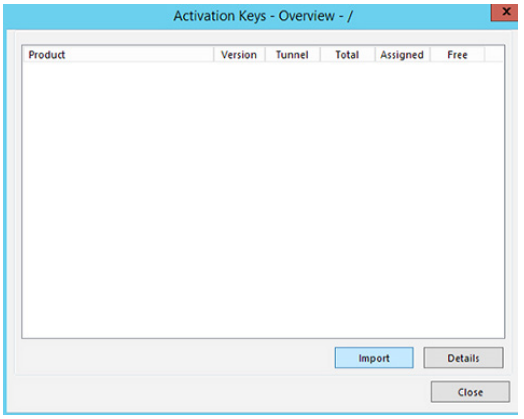
Click "Enable all" and then click on OK.

Now click “License Management” in the same place you selected “Client Configuration” and repeat the steps. Then click “Close.” Now exit the NCP Management Console and then start it again (to load the newly uploaded plug-ins).

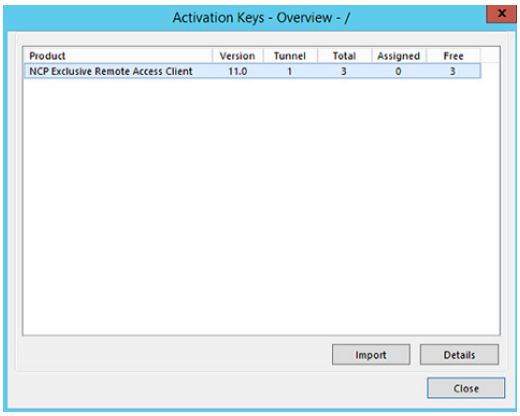
Before starting to create the client configuration you need to load the client license into the Management Server.



Click the “Edit” tab and then “Activation Keys – Overview.”

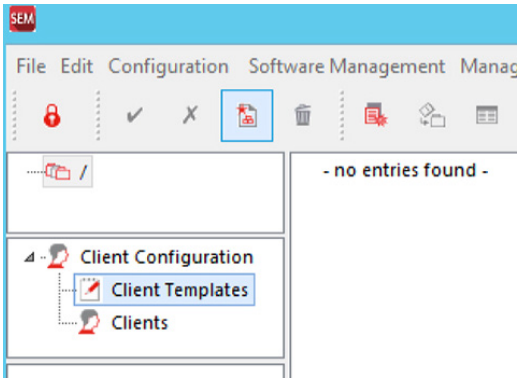


Click “Import” and choose the license file provided by NCP. Load it.

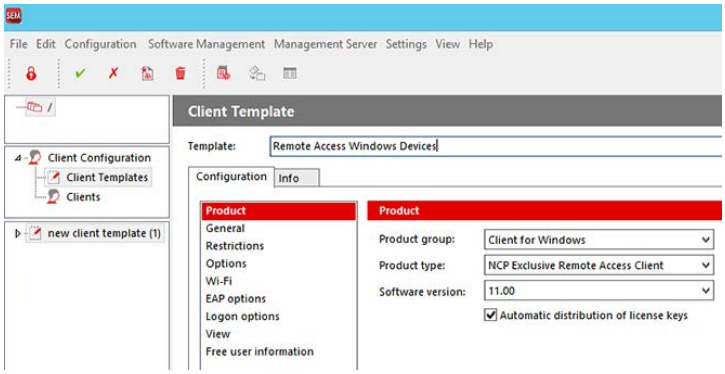


When the load of the license file is complete, it should look like this. Then click Close.

Now we should start to create the configuration template for the clients.



Highlight "Client Templates" then click the blue box icon in the menu bar, or "New Entry."



Give the template a name, here, we used *Remote Access Windows Devices*.

In the Client Template Configuration tab, highlight "Product" and choose the following:

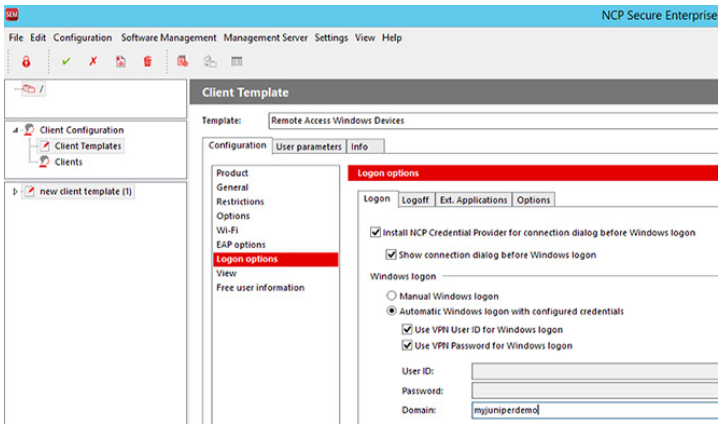
Product group: "Client for Windows"

Product type: "NCP Exclusive Remote Access Client"

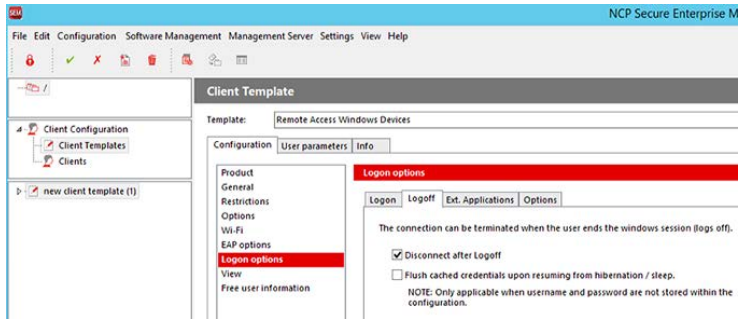
Software version: Pick the latest

Check: " Automatic distribution of license keys."

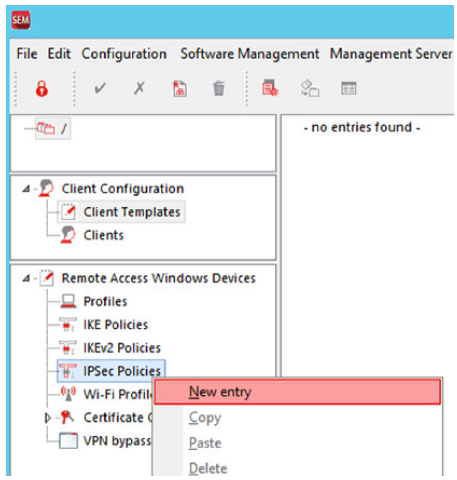
Finally, click on "new client template" in the left sidebar. Click "Save" when asked.



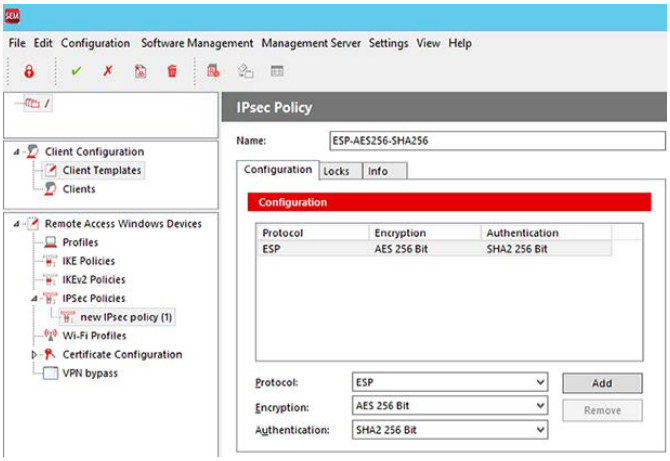
Highlight "Logon options > Logon tab" and check all the items as shown here. This allows the user to establish a tunnel and automatically log on to Windows. (The domain represents your Active Directory domain.)



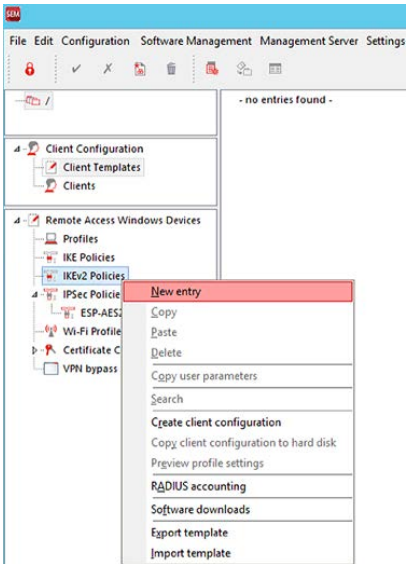
Highlight "Logon options > Logoff tab, and check "Disconnect after Logoff." Then click "new client template" in the left sidebar and click "Save" when asked.



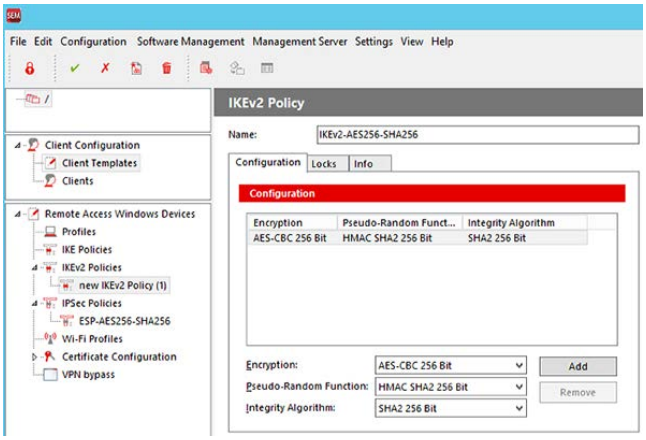
Right-click "IPsec Policies" in the sidebar and click "New entry."



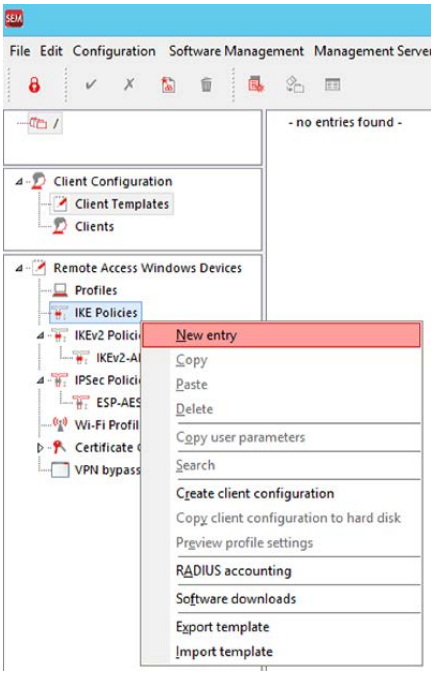
Give this new policy a name. Here, we use “ESP-AES256-SHA256” so we know what functions are used. Then click “new IPsec policy (1)” in the left sidebar menu, and click “Save” when asked.



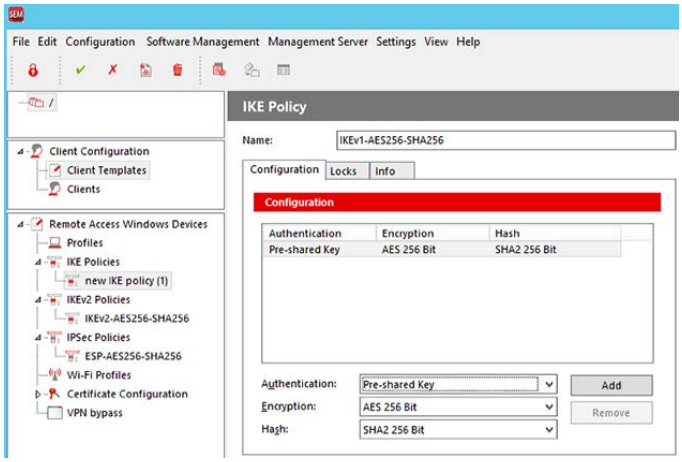
Right-click “IPsec Policies” in the sidebar and select “New entry.”



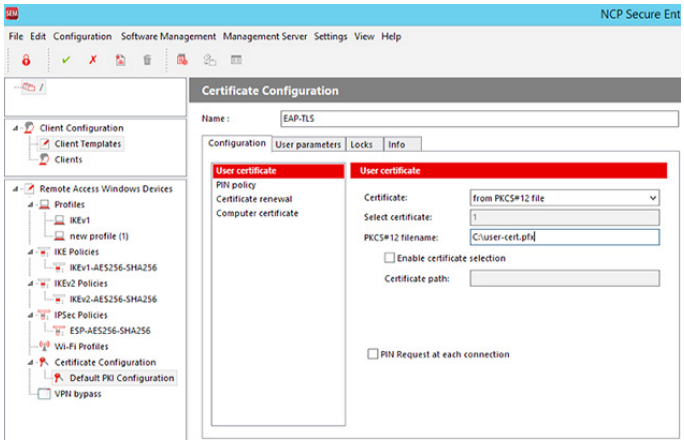
Give this new policy a name. Here, we use "IKEv2-AES256-SHA256" so we know what functions are used. Then click "new IKEv2 policy (1)" in the left sidebar menu, and click on "Save" when asked.



Right-click "IKE Policies" in the sidebar and then select "New entry."

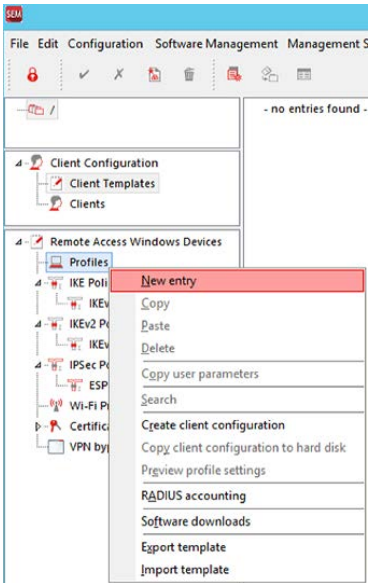


Give this new policy a name. Here, we use “IKEv1-AES256-SHA256” so we know what functions are used. Then click “new IKE policy (1)” in the left sidebar, and click on“Save” when asked.

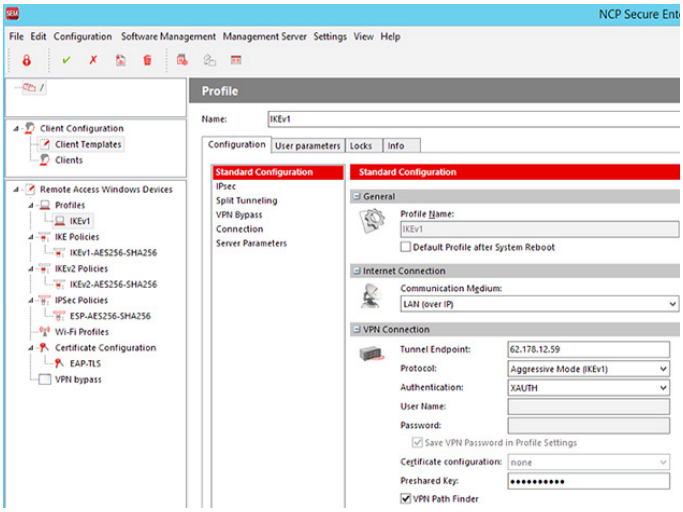


First click on “Certificate Configuration” in the sidebar menu and then highlight “Default PKI Configuration” underneath it to show the defaults. Give it a file name and then click “Default PKI Configuration” in the left sidebar again. Click “Save” when asked.

Next we will create two profiles, one for IKEv1 and one for IKEv2 with EAP-TLS.

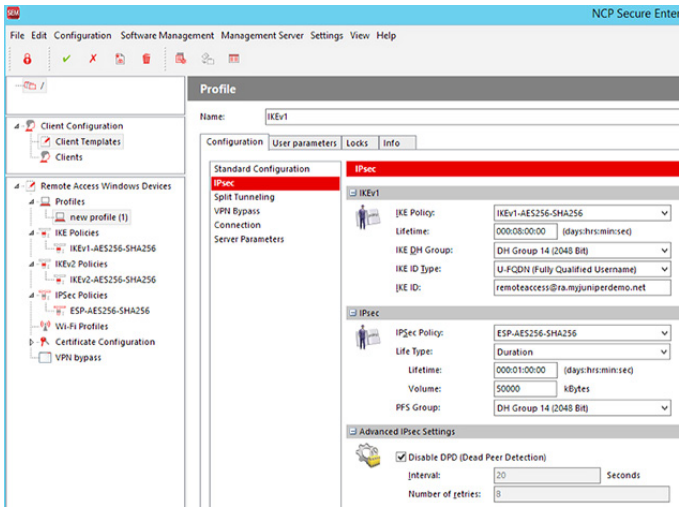


Right-click "Profiles" and then select "New entry."

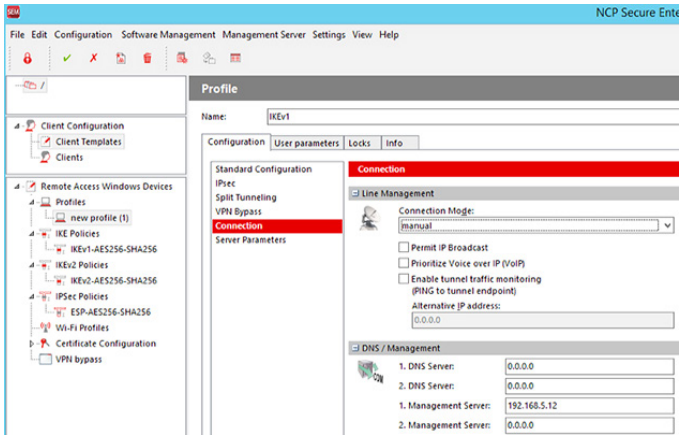


Give this new profile a name. Here, we use "IKEv1" so we know what functions it should be used for. Choose the options shown here. Keep in mind that "Tunnel Endpoint" and "Preshared Key" should be applicable to what you have configured on the SRX.

Highlight “IPsec” in the the left menu under the “Standard Configuration” tab.

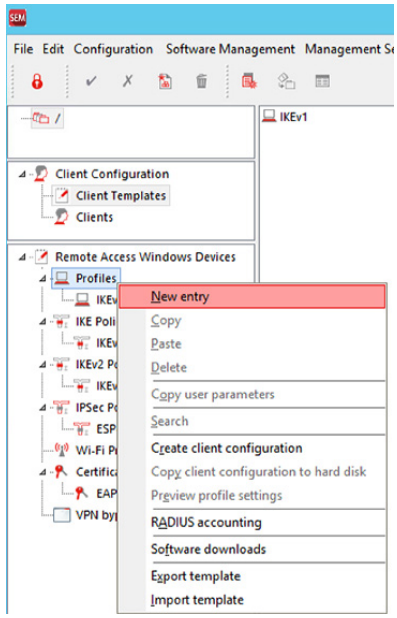


Choose the options shown here under the IKE, IPsec, and Advanced IPsec settings. The click “Connections” in the left menu under the “Configuration” tab.

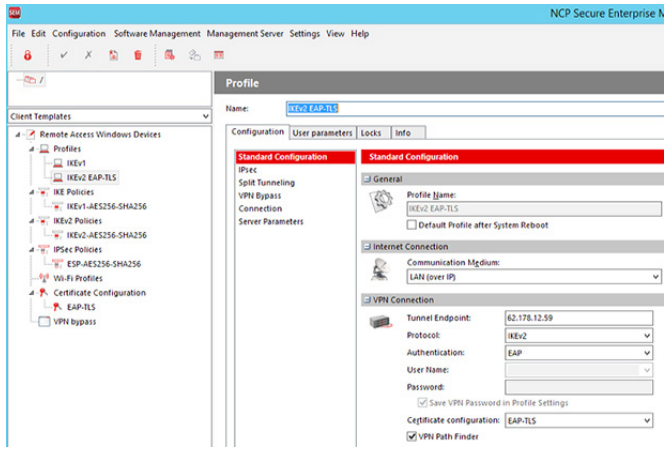


Enter the IP-Address of the NCP “Management Server.”

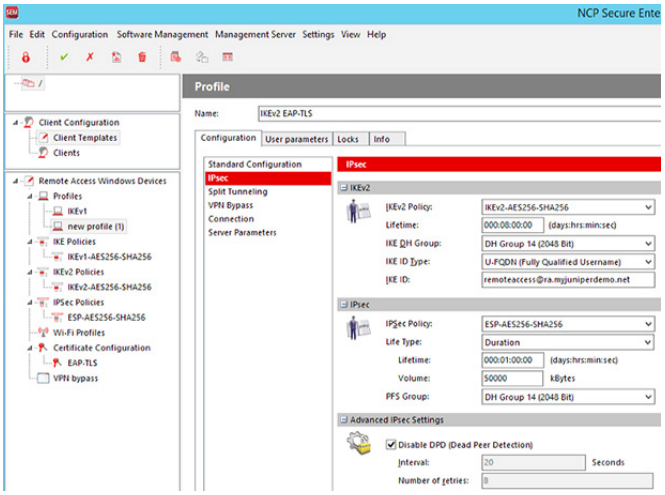
Then click “new profile (1)” in the left sidebar menu, and click “Save” when asked.



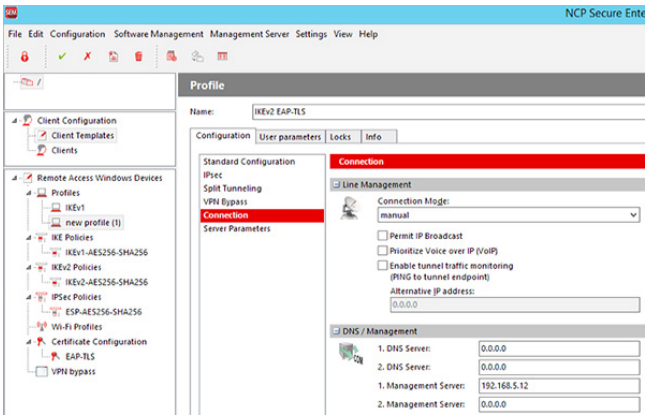
Right-click "Profiles" and click "New entry."



Give this new profile a name. Here, we use "IKEv2 EAP-TLS" so we know what functions it should be used for. Choose the options as shown. Keep in mind that "Tunnel Endpoint" should be applicable to what you have configured on the SRX. Then highlight "IPsec" under the "Standard Configuration" tab.

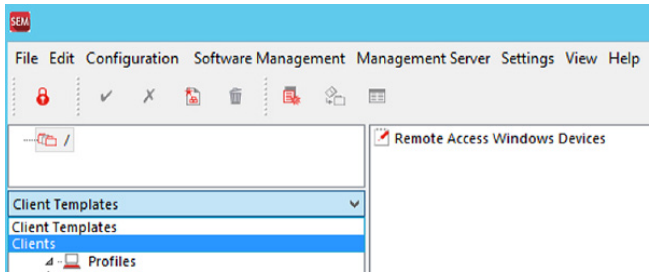


Choose the options as shown for the IKE, IPsec, and Advanced IPsec settings. Then click “Connection” in the menu under the “Configuration” tab.

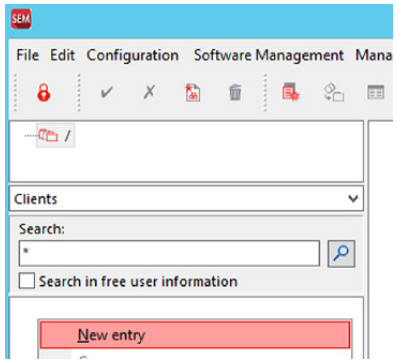


Enter the IP address of the NCP “Management Server.” Then click “new profile (1)” in the left sidebar, and click “Save” when asked.

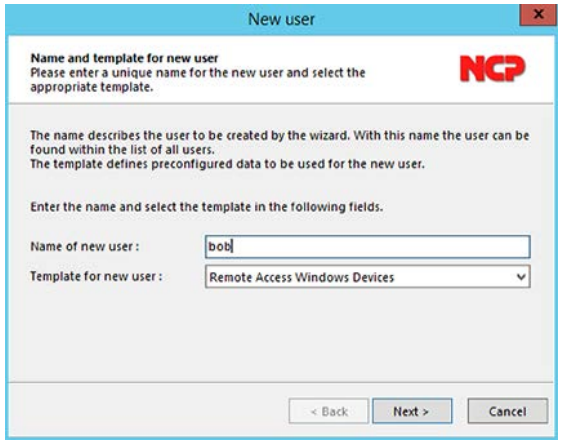
Now you should create a generic user so you can export the configuration.



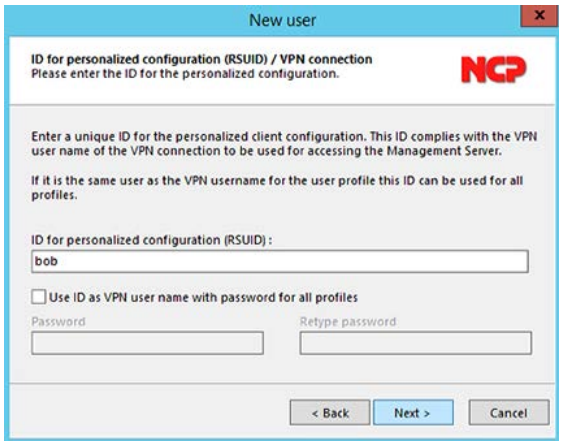
Change from "Client Templates" to 'Clients."



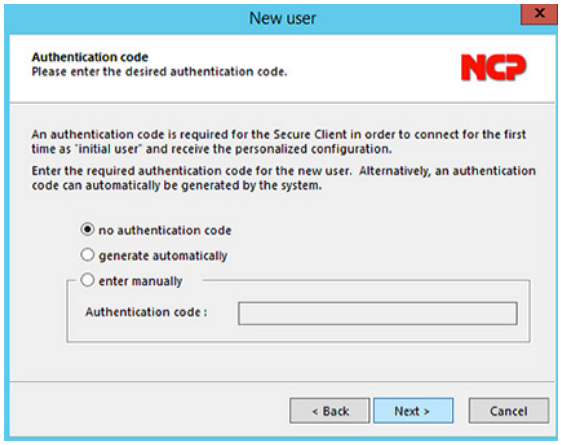
Right-click in the window and then click on "New entry."



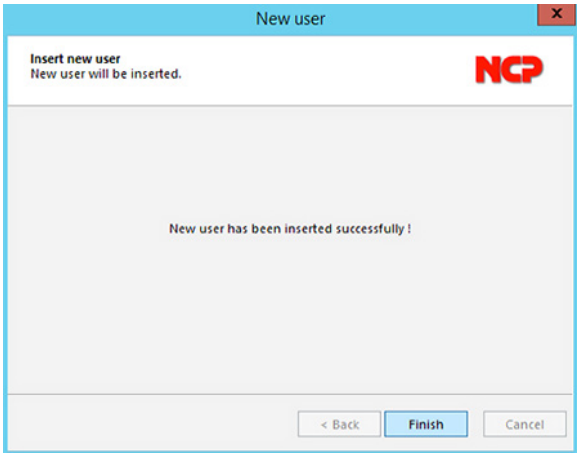
Give this user a generic name. Here we use "bob." (These steps need to be repeted if we add more users.) Then click Next.



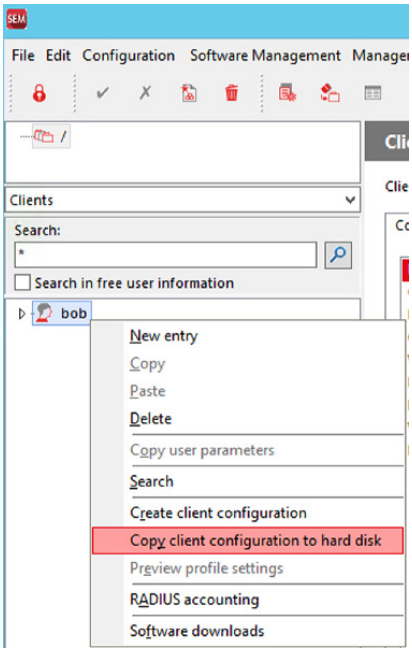
And you must give this other user a name, too, but we will change it in a few steps. Click Next.



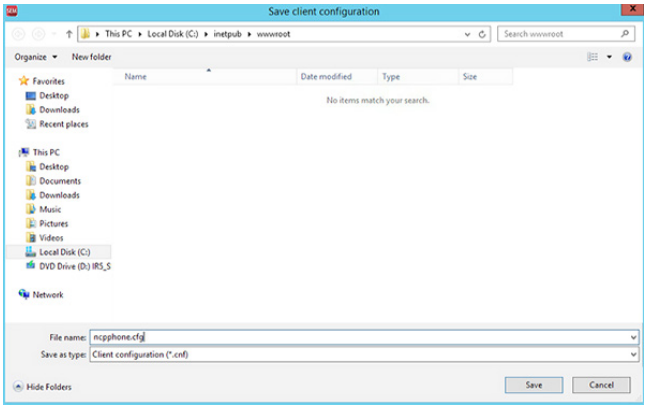
Check “no authentication code” and click Next.



Click Finish.



Now you should export this configuration, so you can use it on all clients. Right-click the "Generic user" then select "Copy client configuration to hard disk."



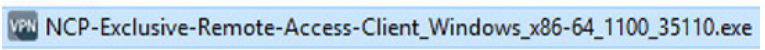
For easy distribution in our lab, we saved it to the “webroot” directory so we can browse for it later.

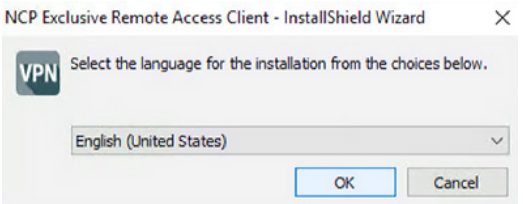
If you deploy more users, one way would be to either create a directory under root for each user, or distribute the file in a different way. Keep in mind that the file must keep the name “ncpphone.cfg” for each user.

This part of the deployment is done.

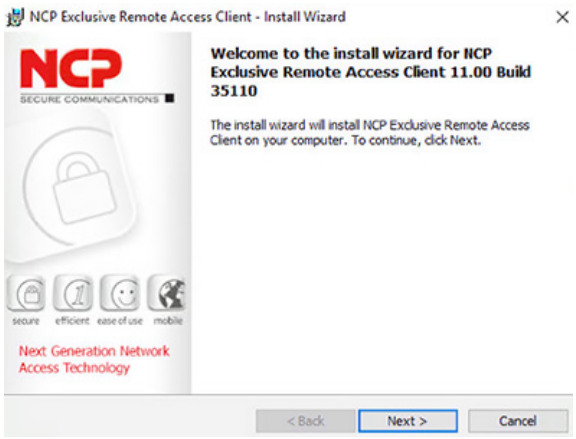
Install NCP Client for Windows

Run the installer.

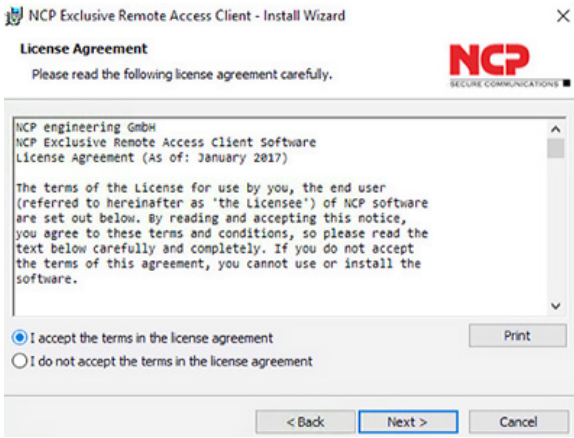




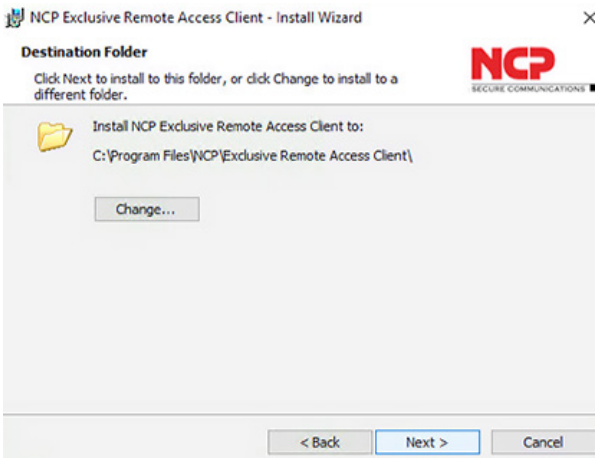
Click OK.



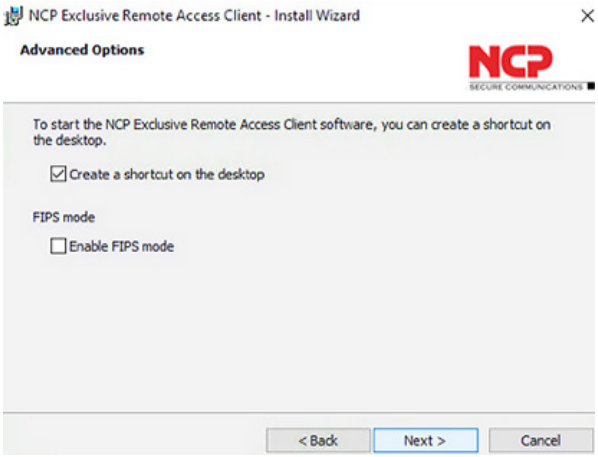
Click Next.



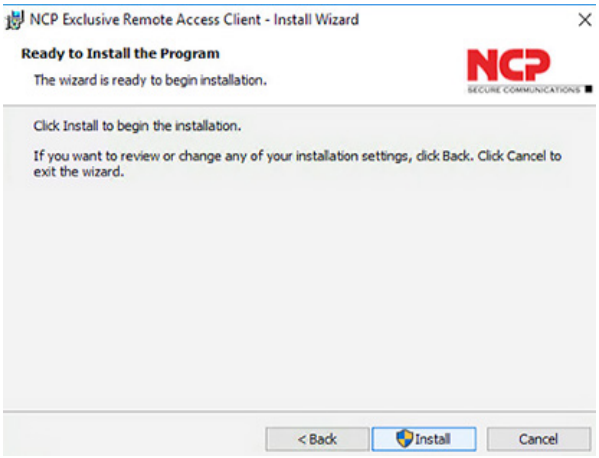
Read the license terms and check "I accept the terms in the license agreement," then click Next.



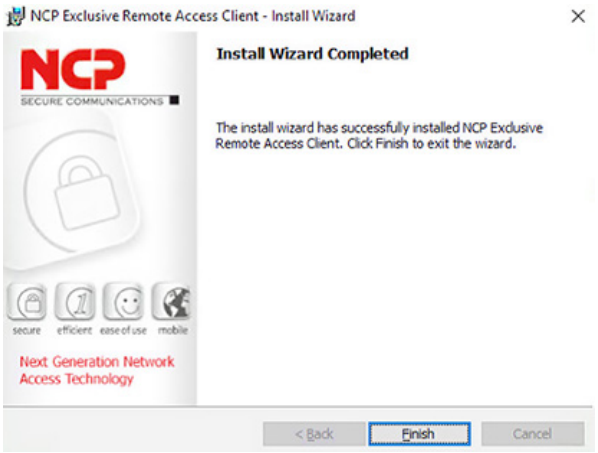
The installation directory will be important soon, remember the location and click Next.



Click Next.



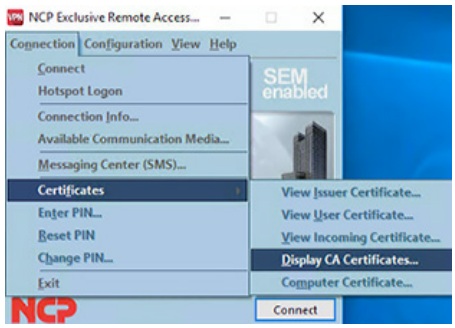
Click "Install" and when asked for username and password, fill them in and then proceed.

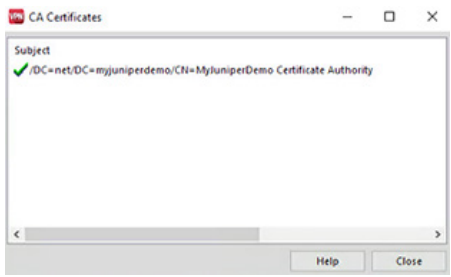


Click Finish when done. Restart the computer.

Optional: Clients that use IKEv2 EAP-TLS most often follow this instruction: copy the CA Root Certificate from the location you saved it to before, to "`C:\Program Files\NCP\Exclusive Remote Access Client\CaCerts\`" or an equal directory if you have the client installed in a different path.

Let's verify that the CA root certificate is installed correctly by clicking *Connection> Certificates> Display CA Certificates*.





This is how it should look (keep in mind that you might have a different path name) and then click Close.

Now close and quit the NCP client.

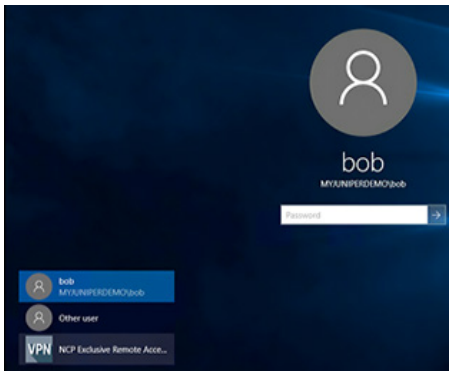
Copy the downloaded ncpphone.cfg file into the installation directory of the NCP client.

Now you have one or two options to establish the tunnel:

1. By clicking “Connect” you will establish your tunnel, enter the correct credentials.

2. When the user starts the computer, an option will be given to automatically establish the tunnel and log in to the network.

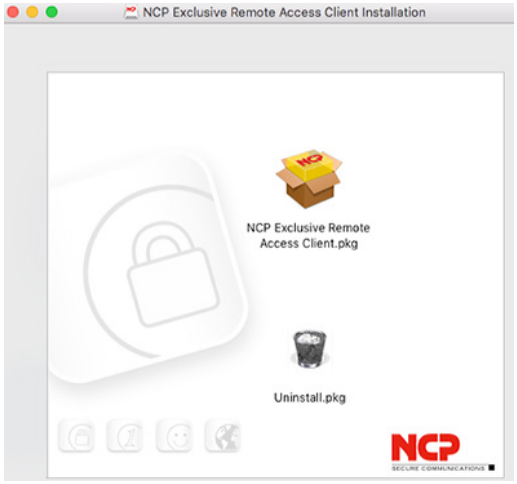
Click “NCP Exclusive Remote Access.”



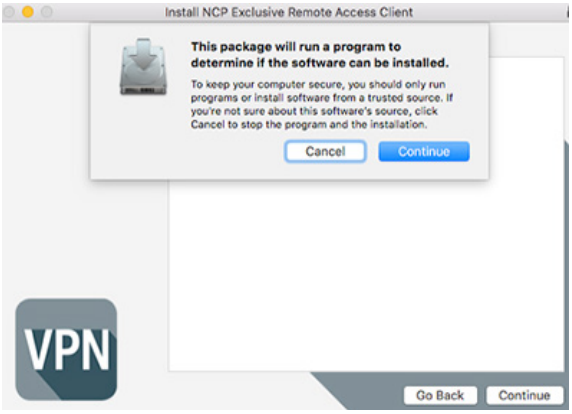
Install NCP Client for Mac OSX

Now let’s cover EAP-TLS for Mac.

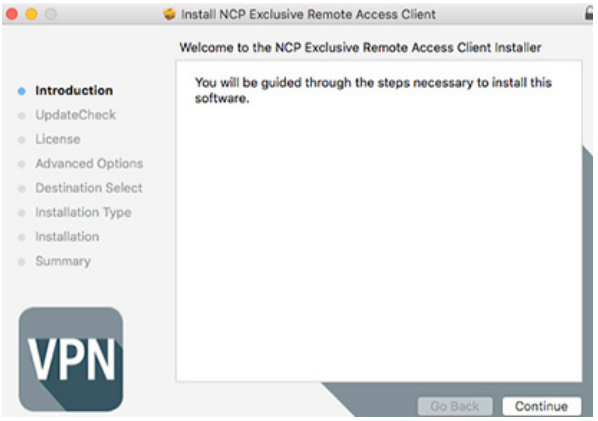
Start the installer.



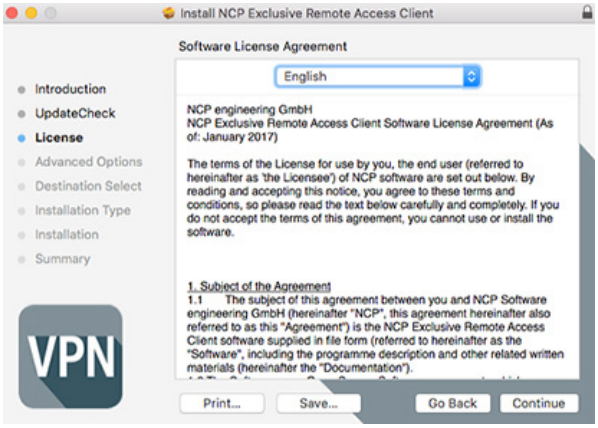
Double-click the “NCP Exclusive Remote Access Client .pkg” icon.



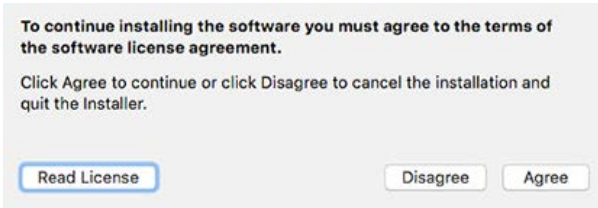
Click Continue.



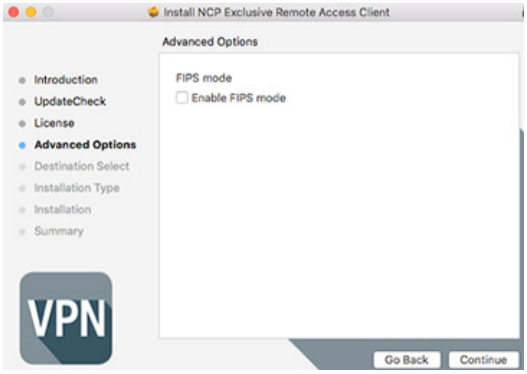
Click Continue.



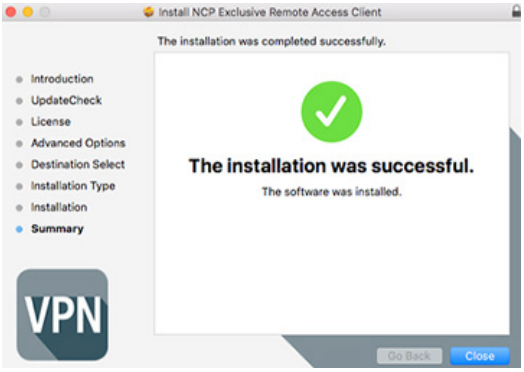
Click Continue.



Read the license terms and accept, then click Agree.



Click Continue.



Click Close.

Now close and quit the NCP client. Copy the downloaded ncpphone.cfg file into the installation directory of the NCP client: *Macintosh HD > Library > Application Support > NCP > Secure Client*.

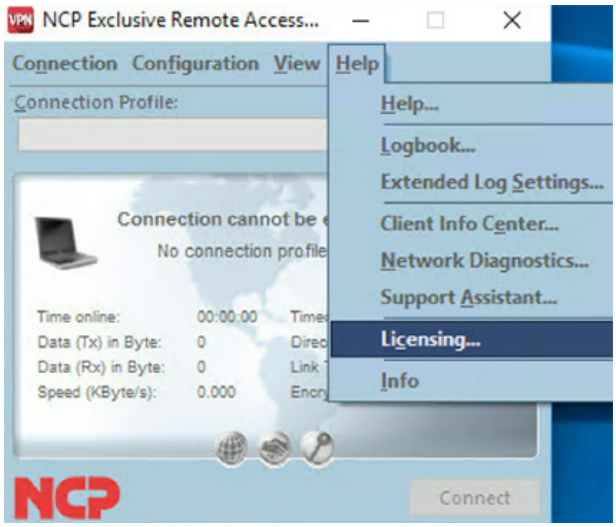
Now establish the tunnel by clicking "Connect"; start to establish your tunnel, and enter the correct credentials.

Verification

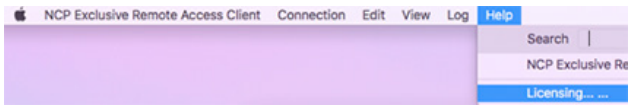
Let's verify that your configuration is working.

First, check that your client has got a valid license, this can takes up to 45 minutes after the client has connected the first time.

In Windows, check here:



In MAC OSX check here:



Some useful syntaxes on the SRX side:

```
user@RemoteAccessDemo# run show security ike active-peer
user@RemoteAccessDemo# run show security tcp-encap connection
root@RemoteAccessDemo# run show security tcp-encap statistics
root@RemoteAccessDemo# run show security ike security-associations
root@RemoteAccessDemo# run show security ipsec security-associations
```

And some useful syntaxes on the Microsoft side:

```
Event viewer -> Custom views -> ServerRoles -> Active Directory Certificate Services
Event viewer -> Custom views -> ServerRoles -> Network Policy and Access Services
```