Juniper Networks | Engineering Simplicity

Juniper Mist™ Access Assurance—Juniper Validated Design (JVD)

Published
2025-06-04

Table of Contents

Juniper Mist™ Access Assurance—Juniper Validated Design (JVD)

Juniper Networks Validated Designs provide you with a comprehensive, end-to-end blueprint for deploying Juniper solutions in your network. These designs are created by Juniper's expert engineers and tested to ensure they meet your requirements. Using a validated design, you can reduce the risk of costly mistakes, save time and money, and ensure that your network is optimized for maximum performance.

About this Document

Overview

This document covers how to successfully implement Juniper Mist Access Assurance as a cloud-based NAC solution for the enterprise. You will learn how the solution works utilizing the RADIUS protocol and how authentication and authorization are performed in the network. The IEEE 802.1X Extensible Authentication Protocol (EAP) methods are used to perform a strong authentication of wired and wireless clients. Unlike traditional designs where the RADIUS server needs to be installed, managed and maintained by each customer on their own, the shift towards a cloud-based service offered by Juniper leads to a scalable design that grows on-demand for the customer. The shift towards cloud-based services is in line with the general trend of using public identity providers (IdP), hence the Juniper NAC design can leverage the information provided by those for authentication and mapping to an authorization profile to be used on a switch or access point (AP) for a client.

This JVD explains the testbed used and methods of various test cases performed by Juniper Networks.

The recommendations section lists the best practices and advice customers should follow for such integration.

The appendix of this JVD contains test cases and configuration examples to allow customers to repeat the integration of Juniper Mist Access Assurance into their own network based on the provided information.

Solution Benefits

IN THIS SECTION

Product overview

Juniper Mist Access Assurance is a cloud-based service that ensures zero trust, identity-based network access and full-stack policy and segmentation assignments with end-to-end user experience visibility. The service delivers a suite of access control functionality with a flexible, yet simple, authorization policy framework for onboarding guests, IoT, BYOD, and corporate devices. Client connections are controlled based on user and device identities, regulating access for devices connecting to the network. Juniper Mist Access Assurance also provides access control services for devices leveraging 802.1X authentication and MAC Address Bypass (MAB) for non-802.1X allow-listed and wired IoT devices.

Product Description

Juniper Mist Access Assurance is a microservices-based, cloud network access control (NAC) service that enables enterprises to easily enforce a zero trust security model. Access Assurance solves many complex challenges associated with traditional NAC offerings by:

- Removing on-premises server hardware
- Providing inherently highly available and resilient services
- Enabling automatic at-run-time feature updates, security, and vulnerability fixes

Access Assurance extends beyond the capabilities of Juniper Mist IoT Assurance, which simplifies onboarding for headless IoT and BYOD devices. With Access Assurance, IT teams can onboard wired and wireless devices with 802.1X authentication or MAB methods, even for non-802.1X devices.

Access Assurance uses hundreds of different vectors to match the identity of the user and device, such as X.509 certificate attributes, user group memberships, device compliance and posture metrics, and location context. These vectors help determine identity-based network admission criteria, such as the network segment or microsegment a device should connect to and the network policy that should be dynamically applied to a user.

Figure 1: Juniper Mist Access Assurance

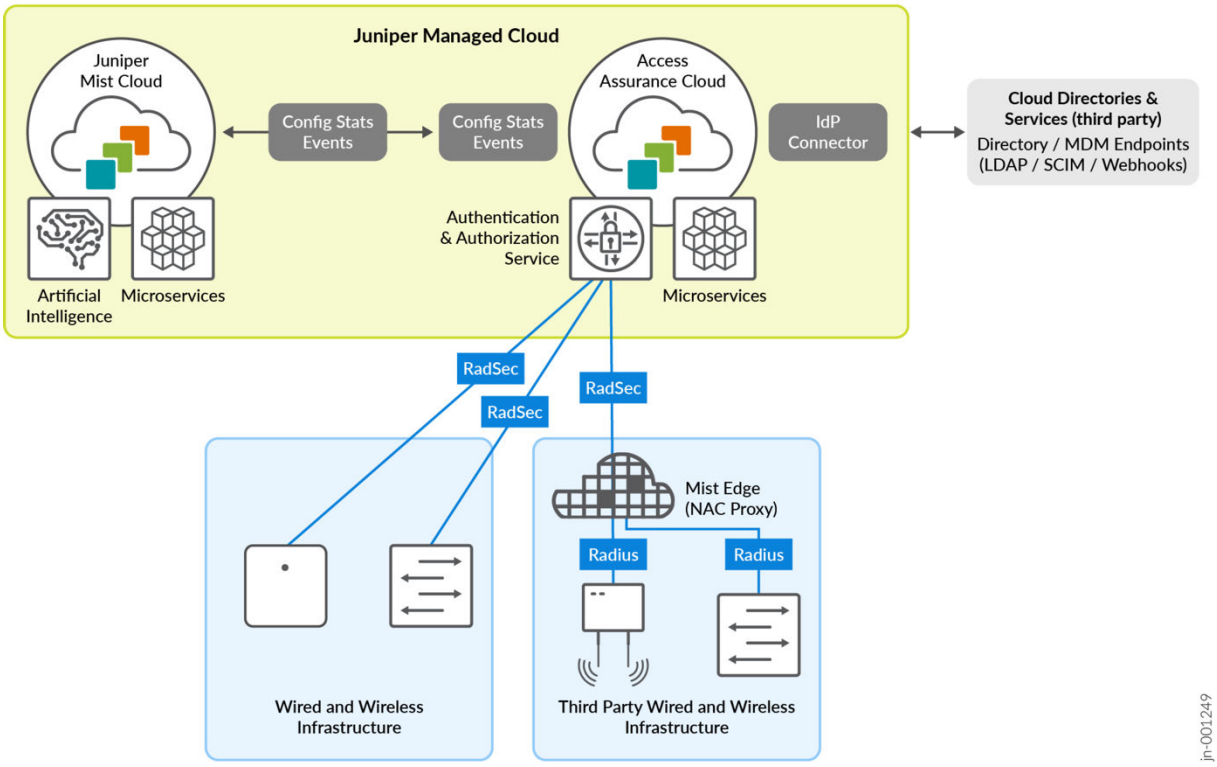
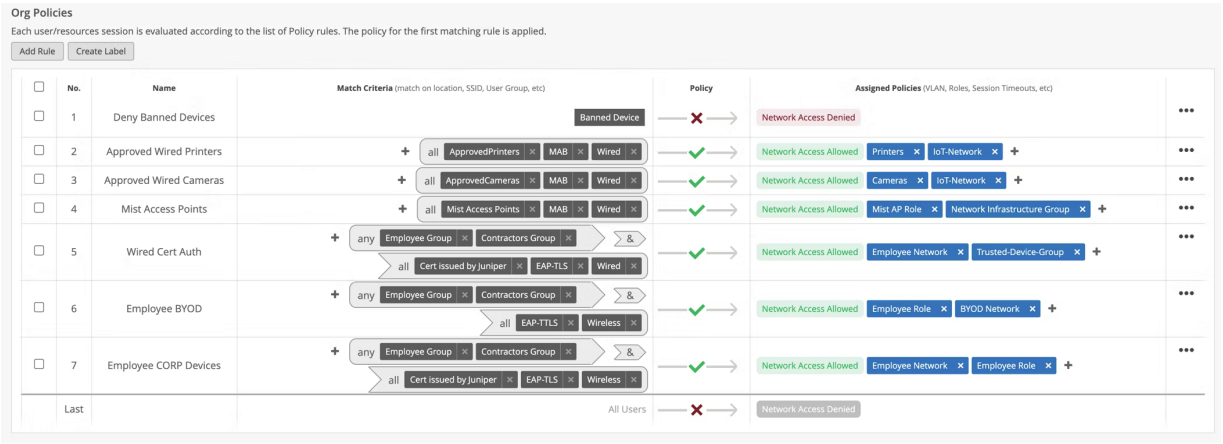
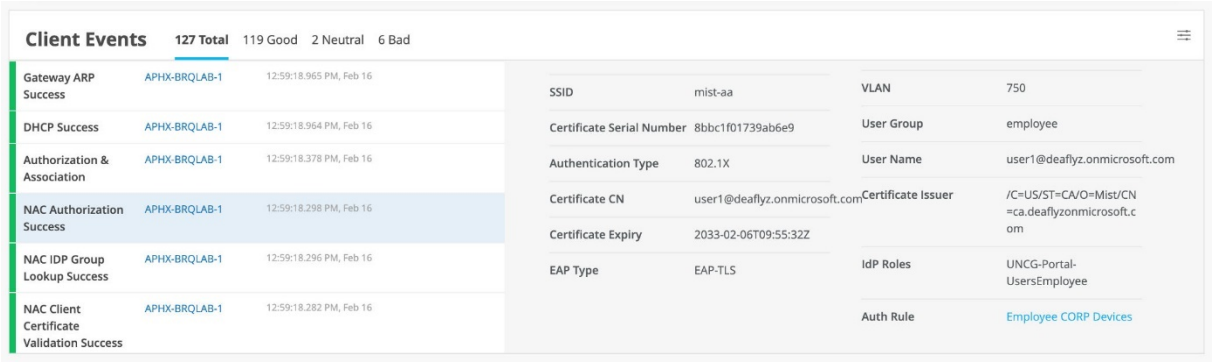


Figure 2: Defining Authentication Policies



Most importantly, Access Assurance provides end-to-end connectivity troubleshooting in a unified view from the client, network infrastructure, and access control perspective, dramatically simplifying Day 2 support. IT admins gain a cohesive view of the end user experience and can determine whether poor experiences are due to client configuration, network infrastructure, authentication, or a service.

Figure 3: Review Authentication Events



Architecture and Key Components

Access Assurance is delivered through Juniper Mist™ cloud and powered by Mist AI™. The microservices architecture ties together high availability, redundancy, and autoscaling for optimal network access across wired, Wi-Fi, and WAN. Using geo-awareness, Access Assurance automatically redirects authentication requests from different regions to the nearest Access Assurance instance to provide minimal latency and the best end user experience.

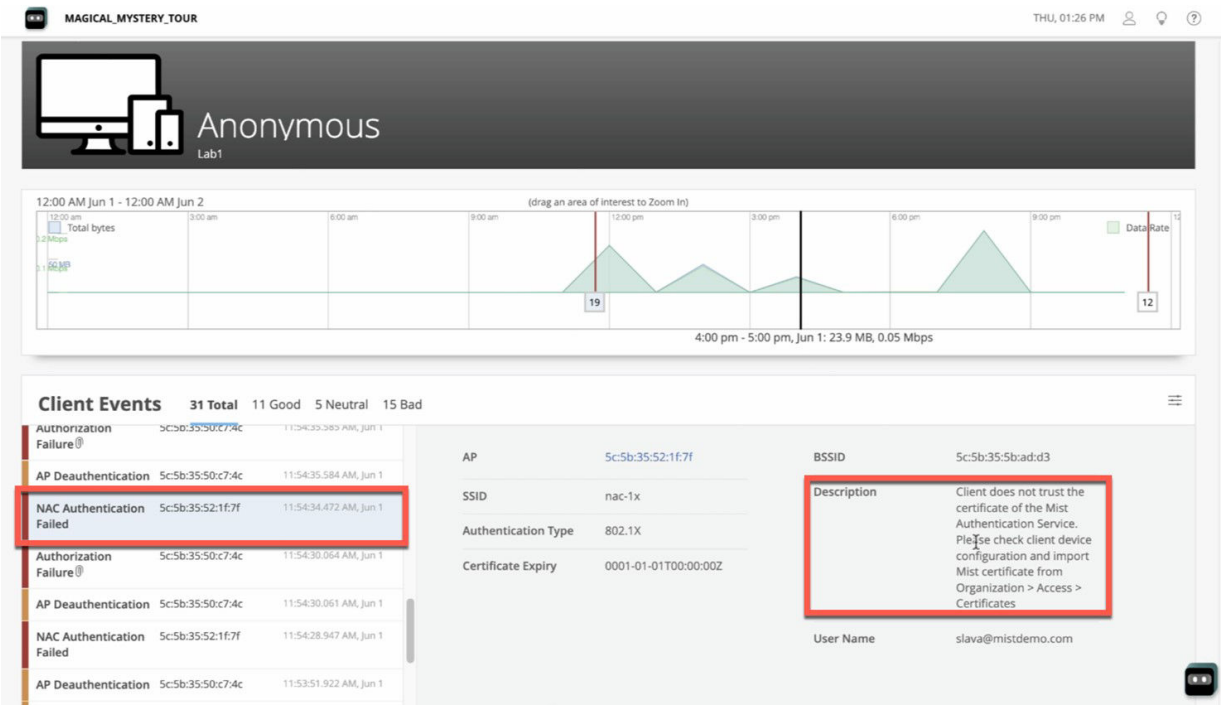
Access Assurance provides an authentication service by integrating external directory services, like Google Workspace, Microsoft Azure AD, Okta Identity, and others. It also integrates external Public Key Infrastructure (PKI) and Mobile Device Management (MDM) providers, such as Jamf, Microsoft Intune, and others, to provide granular user and device identification to enforce identity-based, zero-trust network access control.

Features and Benefits

Prioritizing Client Experiences

Access Assurance provides a unified view of the client connectivity experience and can easily identify a problem and perform root cause analysis. All client events, including connection and authentication successes and failures, are captured by Juniper Mist cloud. With this data, Juniper Mist cloud helps simplify day-to-day operations by easily identifying if an end user connectivity issue is caused by a client configuration mistake, network infrastructure and service problems, or authentication policy configuration issues. The Juniper Mist service-level expectations (SLEs) for wired and wireless clients are enhanced to include network access events, such as authentication events, certificate validations, and more.

Figure 4: Log Message Example

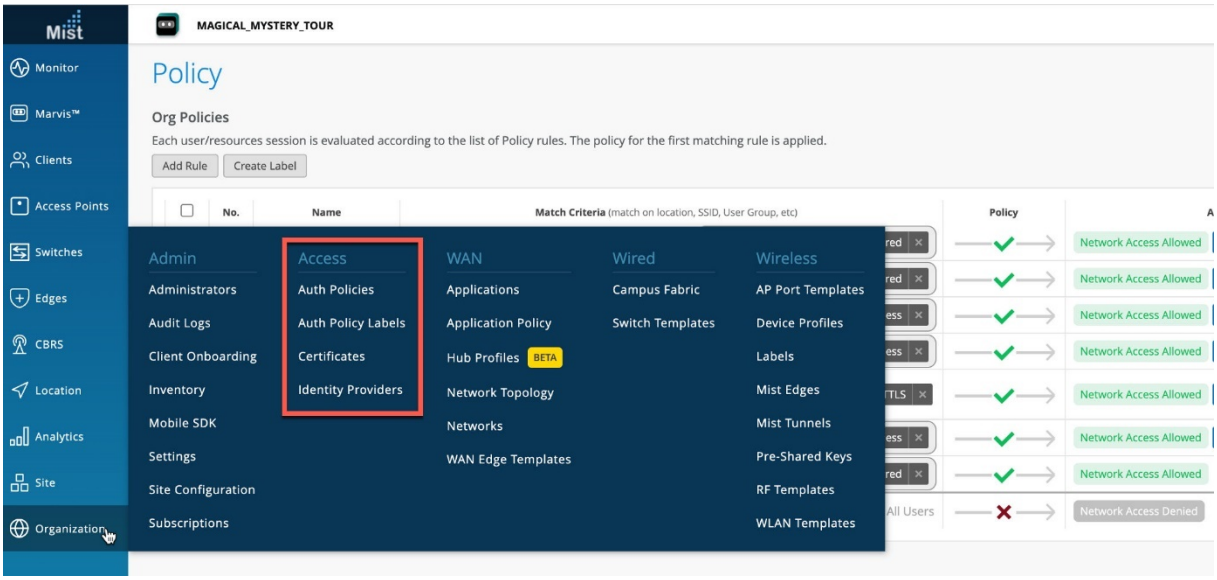


Single Pane of Glass for Management and Operations

Access Assurance is tightly integrated with Juniper Mist cloud, providing full-stack management and day-to-day operations for Wi-Fi Assurance, Wired Assurance, SD-WAN Assurance, and Access Assurance in one dashboard for end-to-end visibility. The Marvis® Virtual Network Assistant leverages data from multiple sources for anomaly detection to provide actionable metrics. Through the dashboard, users can:

- Create and apply access policies that ensure only authorized devices and users are allowed network access
- Assign users and devices to the correct network segment
- Prevent users and devices from accessing restricted resources
- Add and modify certificates and certificate authorities
- Configure identity providers
- Monitor client activity across the organization

Figure 5: Access Assurance Configuration



Granular User and Device Identity

Access Assurance is capable of granular identity fingerprinting based on X.509 certificate attributes. It also uses IdP information like group membership, user account state, MDM compliance state, client lists, and user location for fingerprinting. The resulting user and device fingerprint provides an identity vector for accurate policy assignment within the zero trust principles.

Figure 6: Credential Types

Network Policy Enforcement and Microsegmentation

Based on user and device identity, Access Assurance can instruct the network to assign a user to a specific network segment (VLAN or a group-based policy tag), as well as enforce network policy by assigning a user role. Such roles can be leveraged in the Juniper Mist WxLAN policy framework or switch policies.

Figure 7: Assign a Dynamic GBP-Tag as Authorization Attribute

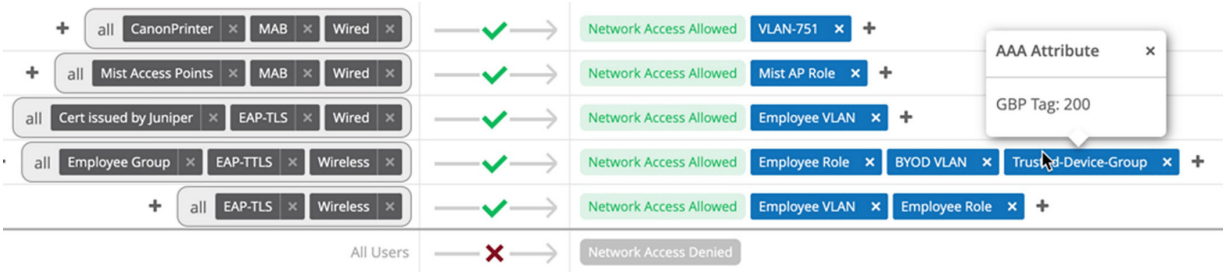
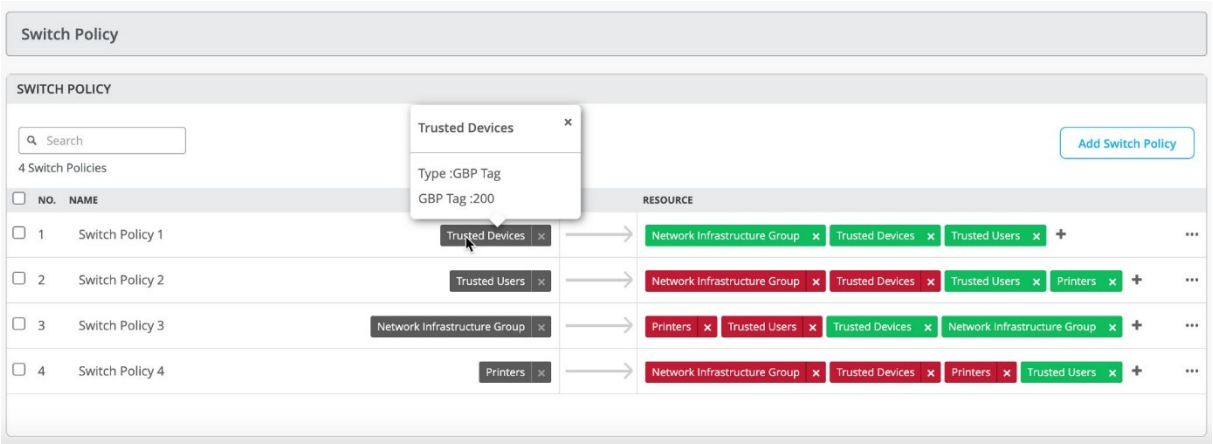


Figure 8: Switch Policy for Applied GBP-Tag to Use



Built-in High Availability and Geo-Affinity

With Access Assurance, organizations gain reliable and low-latency network access control of their networks in single and multi-site deployments. Juniper Networks has deployed cloud instances of its network access control cloud service in multiple regional locations. In multisite deployments, authentication traffic coming from the network infrastructure is automatically directed to the nearest Access Assurance instance. Latency is minimized and users enjoy an exceptional wireless experience. This automated process is fully transparent to users and requires no involvement from the IT team. Organizations are assured reliable, redundant network access for client devices, regardless of the state of the nearest regional instance.

Automatic Feature and Security Updates

The Juniper Mist microservices-based cloud architecture keeps Access Assurance optimized with the most advanced technologies. New features, security patches, and updates are automatically added to Access Assurance on a bi-weekly basis without interruptions or service downtime. This capability dramatically simplifies and improves service operations for network IT administrators, eliminating lengthy software upgrades and service downtime. Juniper can easily deploy new features and functions to its cloud-based services, bringing advancements to market more rapidly and continuously improving your client-to-cloud experience.

Access Assurance Extends Juniper Mist IoT Assurance

Access Assurance is paired with Juniper Mist IoT Assurance to build out controls for onboarding and management of corporate devices with 802.1X authentication and MAC-less onboarding of non-802.1X IoT and BYOD devices. IoT Assurance capabilities simplify IT operations and secure connections for headless IoT and BYOD devices using a Multiple Preshared Key (MPSK) mechanism. It incorporates a full suite of access control functionality leveraging MPSK or Private Preshared Key (PPSK) as a new type of identity and policy vector.

IoT Assurance also provides preshared key (PSK) portal creation, enabling BYOD onboarding workflows by automating PSK generation based on user identity, leveraging Security Assertion Markup Language (SAML) for an SSO experience. It enables seamless client device onboarding using mobile QR codes or by typing a personalized passphrase without installing any client software.

Access Assurance subscriptions include IoT Assurance functionality for simple access control for all clients and devices on your network, no matter how they connect.

Marvis Virtual Network Assistant

Marvis Virtual Network Assistant uses Mist AI to help IT teams interact and engage with their networks. The Marvis AI engine binds together Access Assurance with other Juniper Mist cloud-based services, such as Wired Assurance, Wi-Fi Assurance, and WAN Assurance, helping the operations team move closer to achieving The Self-Driving Network™ with simplified troubleshooting and performance analysis.

Using features powered by Mist AI, help desk staff and network administrators can simply ask a question in natural language and get actionable insights using the Marvis Conversational Interface that helps them identify and solve network issues. Marvis brings proactive anomaly detection into the SLE dashboard. With Marvis Actions, staff gain proactive, actionable insights to identify network access issues across the full stack, providing recommendations for user connectivity issues. This provides our customers with easy root cause analysis across the full network stack and authentication services.

API-Driven Architecture

Access Assurance service is 100% based on public Representational State Transfer (REST) APIs that allow easy integration with external security information and event management (SIEM) or IT service management systems or other platforms for both configuration and policy assignment. These APIs provide the capability to invoke actions based on user or external events, as well as for using the cloud-native Webhook framework. Overall, the Juniper Mist platform is 100% programmable, using open APIs, for full automation and seamless integration with complementary Juniper Networks access, wired, wireless, WAN, security, user engagement, and asset visibility solutions.

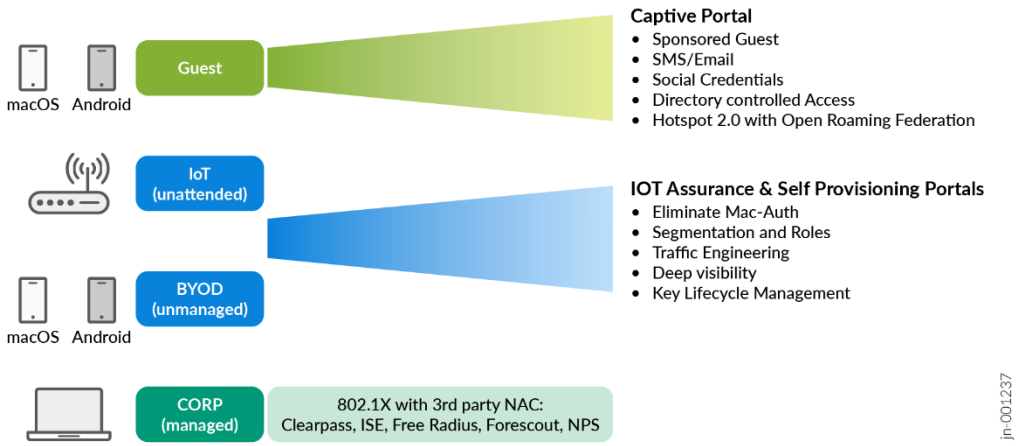
Use Case and Reference Architecture

IN THIS SECTION

Use Case Overview

Enterprise networks can be accessed through various ways by various client types. This triggers the need for some kind of federation of how you manage and grant access to those devices and define which resources of the network each client can use.

Figure 9: Network Access Use Cases



Architecture: Why Use the RADIUS Protocol?

Today, all central enterprise authentication schemas are based on the usage of the RADIUS protocol and using IEEE 802.1X for strong client authentication using various EAP methods.

There are two other authentication protocols sometimes used in the enterprise:

- TACACS and TACACS+ can be considered as legacy protocols. Today, they're still sometimes used for administrative CLI access on a device but that can be done with RADIUS as well. Do not use it in new deployments. Juniper Mist Access Assurance does not support this protocol.

- Diameter is an evolved protocol that is more complex but adds a lot more features for protocol routing and robustness of the messaging. The main usage of the protocol is in mobile networks to replace the legacy SS7 protocol. Juniper Mist Access Assurance does not support this protocol.

The RADIUS Protocol is defined in IETF RFC 2865 and RFC 2866. It serves three main purposes:

- **Authentication**—of users and devices wishing to access the network. The authentication can be just a simple username and password check, or some certificate-based checking of rolled out PKI, checking of a Smartcard or mobile SIM and many more. Hence, the protocol itself is extensible for new authentication methods.

- **Authorization**—of users and devices after they have been successfully authenticated. As part of the authentication, there will be a decision made if a user or device can access the entire network without any restrictions or if access to the network is restricted in some way. What this restriction means depends on what access you can limit and how. The most common example is to assign a VLAN to a wired or wireless client in an enterprise network because the assigned VLAN limits what resources the user can access.

- **Accounting**—of users and devices. After the authentication has been performed and the authorization has been applied, the network will periodically provide statistics about the client's usage of the network, such as time on the network and transferred bytes in and out. This has more relevance in a service provider network for billing purposes and is less commonly used in enterprise networks.

The following items are critical to understanding the RADIUS protocol. We are not reviewing the full details of the protocol and how it is implemented here, however. Consider reviewing the protocol's original RFC or its Wikipedia Article if you need more information.

- The protocol uses UDP messages for exchange. If messages have been lost in transit, there is an automatic re-transmission happening a handful of times until a request is discarded.

- The RADIUS protocol is extensible by nature. The default Attribute Value Pairs (AVPs) are defined in the RFC but vendors can create new AVPs specific to their devices. Usually, such extensions are made to share more information about a device or enhance the authorization parameters to provide more granular enforcement points.

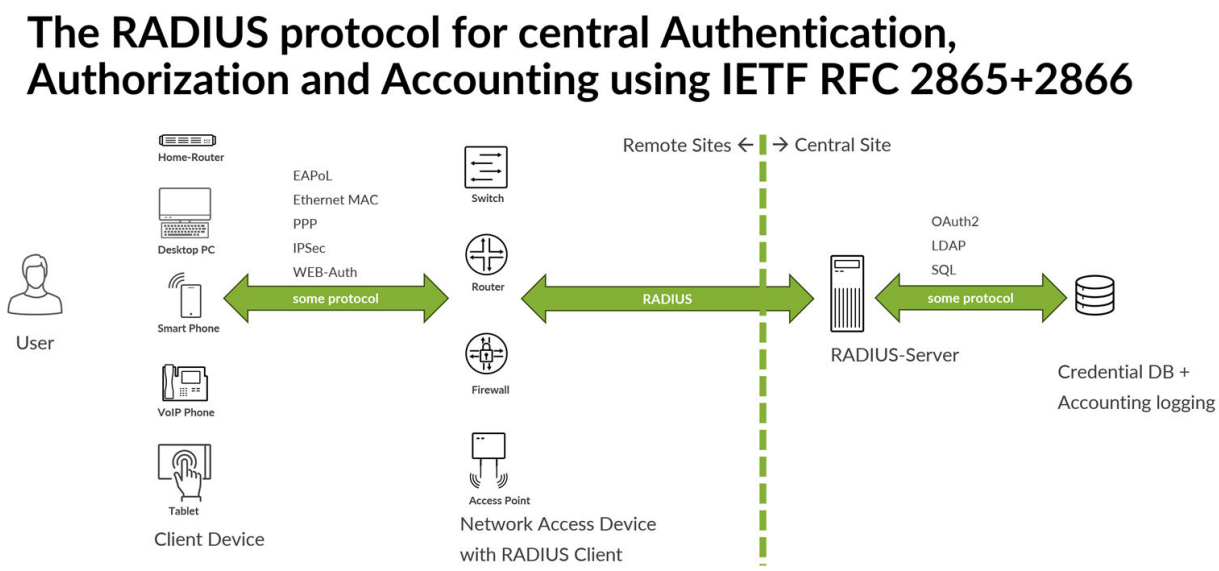
- The network access device is located at the edge of the network at the point of Traffic ingress:
  - In an enterprise network, this is an access switch or an AP.
  - It can be multiple instances in a remote branch or at the lowest level of a campus fabric.

    - The protocol a device uses when attaching to the network access device is not defined. Most commonly you will see:
        - Appearance of a new MAC address on an ethernet switch port.
        - A wired or wireless client starting an EAP authentication (more later in this chapter).
    - The enforcement of authorization of a user or client usually happens at this level.
    - The network access device has a RADIUS client implemented that encapsulates all messages into RADIUS UDP messages and sends them to the RADIUS server.
- The RADIUS server is typically located at a central location of the network and is reachable by the network access devices using a VPN (or public IP address) responding to the RADIUS client messages passively.
    - For redundancy reasons, there are typically two RADIUS servers deployed. However, the network access device chooses when to failover to another RADIUS server.
    - If a vendor has extended the standard RADIUS attributes, the vendor-specific attributes must be configured as well via a RADIUS dictionary from the vendors so that these RADIUS AVPs can be utilized.
    - The RADIUS server is the decision making point in the network for checking the authentication of a client.
        - Getting client credentials of some kind forwarded via the network access device.
        - Retrieving client credentials also from a credential database.
        - Comparing the two sets of information and deciding if they are valid, hence the client can pass or not.
    - The RADIUS server either has a local, built-in credential database or a credential database that can be accessed remotely using a standard protocol. These protocols are typically one of the following:
        - OAuth2 when authenticating using a public IdP.
        - Lightweight Directory Access Protocol (LDAP) when a local (Active) Directory is used.
        - SQL or HLR/HSS databases are more typically used by (mobile) service providers.
    - Along with the user record, the local or remote credential database also has information about the authorization profile to be used upon successful authentication of a client.
    - When authentication for a user completes successfully, the RADIUS server responds with an Access-Accept message that also can contain:

- Dynamic cryptography key material to be used between the client device and the network access device further on to establish secure communication.
- Authorization RADIUS AVPs to be enforced on the network access device to limit the access the client has to the network. Using vendor specific AVPs, the RADIUS server will only send those which are understood on the network access device.

Figure 10 on page 13 shows where the RADIUS protocol is implemented in the network and how it is used.

Figure 10: Implementation and Use of the RADIUS Protocol



The example workflow in Figure 11 on page 15 uses MAC address-based authentication at a switch to review how the framework works together.

1. A device wishing to access the wired network sends some messages (ARP/DHCP request).
2. The switch sees a new MAC address on the port and blocks all further communication to the network.
3. The newly detected MAC address is converted into a username and password and sent as a RADIUS Access-Request for approval. Other RADIUS AVPs are embedded as well, such as one describing the location (port information) about the request.
4. Upon receiving the RADIUS access request, the RADIUS server analyzes the content of the RADIUS message and extracts the RADIUS AVPs needed to perform the authentication. In our case, the username attribute is used, and a request is sent to an internal credential database to receive information about the username (which is identical to the MAC address of the client).

5. The credential database returns either an empty password if the user record has been found or, if valid, a password and an authorization profile (usually a string) to be used for this client.

6. The RADIUS server compares the passwords sent from client and the credential database. If they are a match, an Access-Accept message will be created as a response to the switch. As part of the successful authentication, the authorization profile is used to add the RADIUS AVPs stored for a particular network access device (the switch) and add them as well to the Access-Accept message. In our case, we want the client to use a particular VLAN, hence, the attribute “Tunnel-Private-Group-ID” contains the VLAN name on the switch configured as reference.

7. The RADIUS server returns the Access-Accept as well as the authorization information (VLAN to be assigned) back to the client.

8. The switch, acting as the network access device, upon receiving the Access-Accept message, allows the client to communicate further with the network by unblocking the port. Should the switch also receive authorization RADIUS AVPs, they are applied as well. In our case, the default VLAN assigned to the port is replaced by the one we have authorized the client to use.

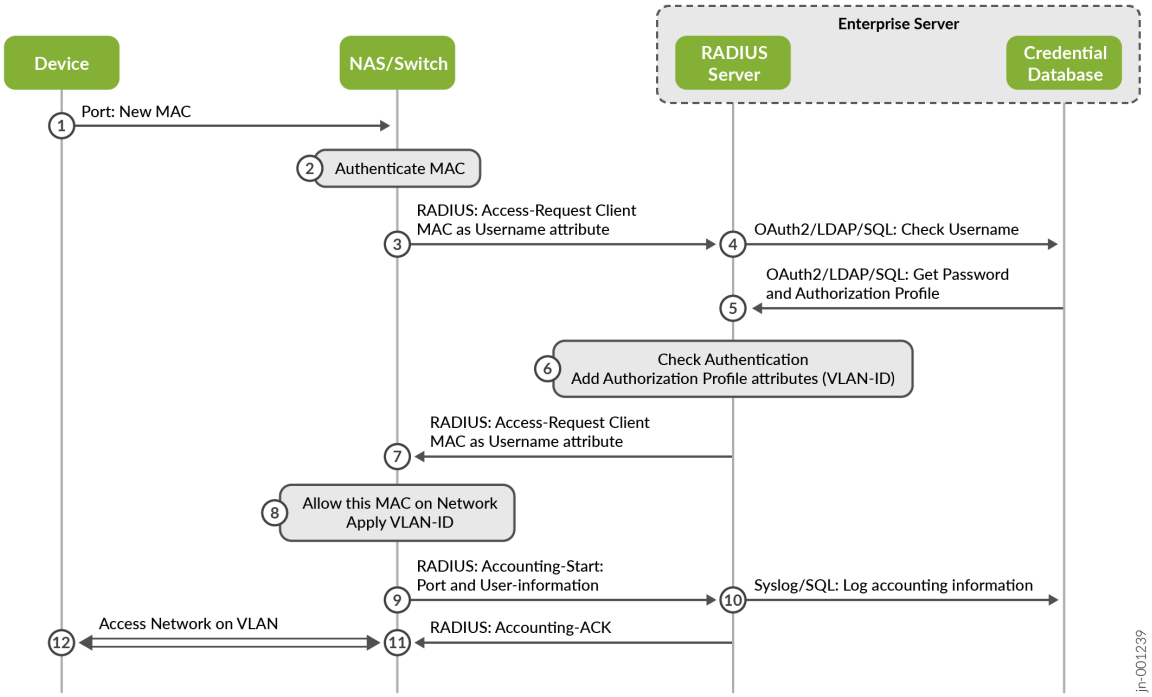
9. Optional: A RADIUS Accounting-Start message is created and sent to the RADIUS server. Again, this is not required in enterprise networks.

10. Optional: The RADIUS server may store locally or forward this information in a database of some sort.

11. Optional: The RADIUS server will acknowledge the accounting message.

12. The client can now access the network. If authorization has been used, the switch will enforce it and limit access in some way.

Figure 11: A Typical Authentication Process Example



NOTE: In our example, a simple MAC address-based authentication (MAB) has been used. However, a known MAC address may easily be reused by an attacker. Therefore, MAB should only be used if the client does not support 802.1X authentication.

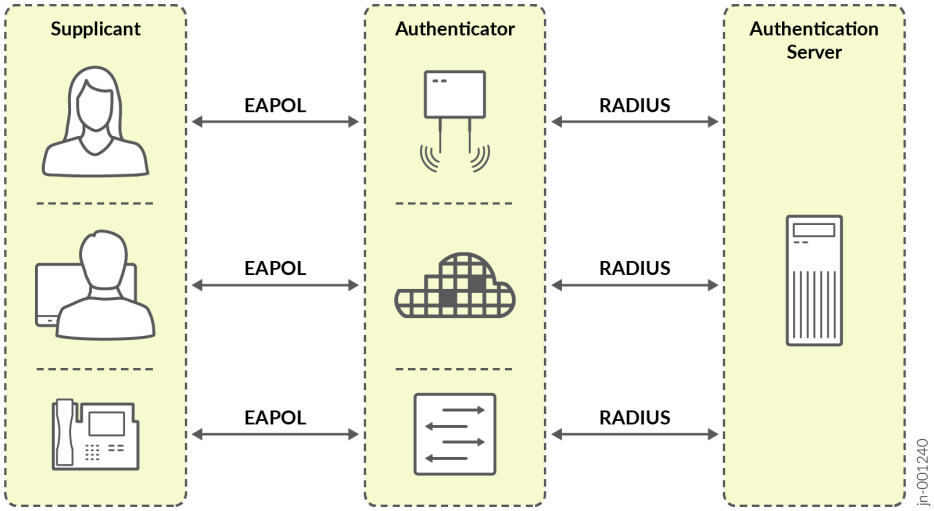
Architecture: 802.1X Authentication: What Is It?

- Standard for port-based network access control.
- Can provide very strong cryptographic authentication.
- Always involves three parties:
  - **Supplicant—**Device that wishes to access the LAN or WLAN. In this JVD, we have the following supplicants:
    - Windows 11 desktop client wired or wireless
    - Linux desktop client wired or wireless

- **Authenticator—**A network access device that provides a data link between the client and the network that can allow or block traffic between the two. In this JVD, we have the following authenticators:
  - Juniper Networks® EX Series Switches
  - Juniper® Series of High-Performance Access Points
- **Authentication Server—**Receives and responds to requests for network access. In this JVD, we have the following authentication server:
  - Juniper Mist Access Assurance

- Defines encapsulation methods of EAP:
  - The EAP methods used are exchangeable. They are negotiated between the supplicant (which offers the different encapsulation methods it supports one-by-one) and the authentication server (which acknowledges a method offered by the supplicant that it supports).
  - Most common transport between the supplicant and the authentication server is EAP over LAN (EAPoL). From NAS to authentication server, those messages get tunneled inside RADIUS.
  - The supplicant selects the outer EAP identity to be used called the Network Access Identifier (NAI) as described first in RFC 2486. This allows the receiving authentication server to proxy the request to another authentication server in case it is not responsible for a roaming client.
  - The NAS/authenticator is NOT involved in the authentication process itself. That process is only handled between the supplicant and the authentication server. The NAS/authenticator only forwards messages. This means that if the authenticator supports 802.1X, it will be capable of handling all the EAP protocols known. The question is always whether the supplicant and the authentication server both support a given EAP protocol.
  - Based on the EAP protocol used, the Access-Accept message sent by the authentication server to the authenticator can contain dynamically created cryptographic key material. This can then be used to establish an encrypted link between the supplicant and the authenticator. The most commonly used is enterprise-grade dynamic WPA2 and WPA3 encryption for WLAN access.

Figure 12: EAP Authentication Framework

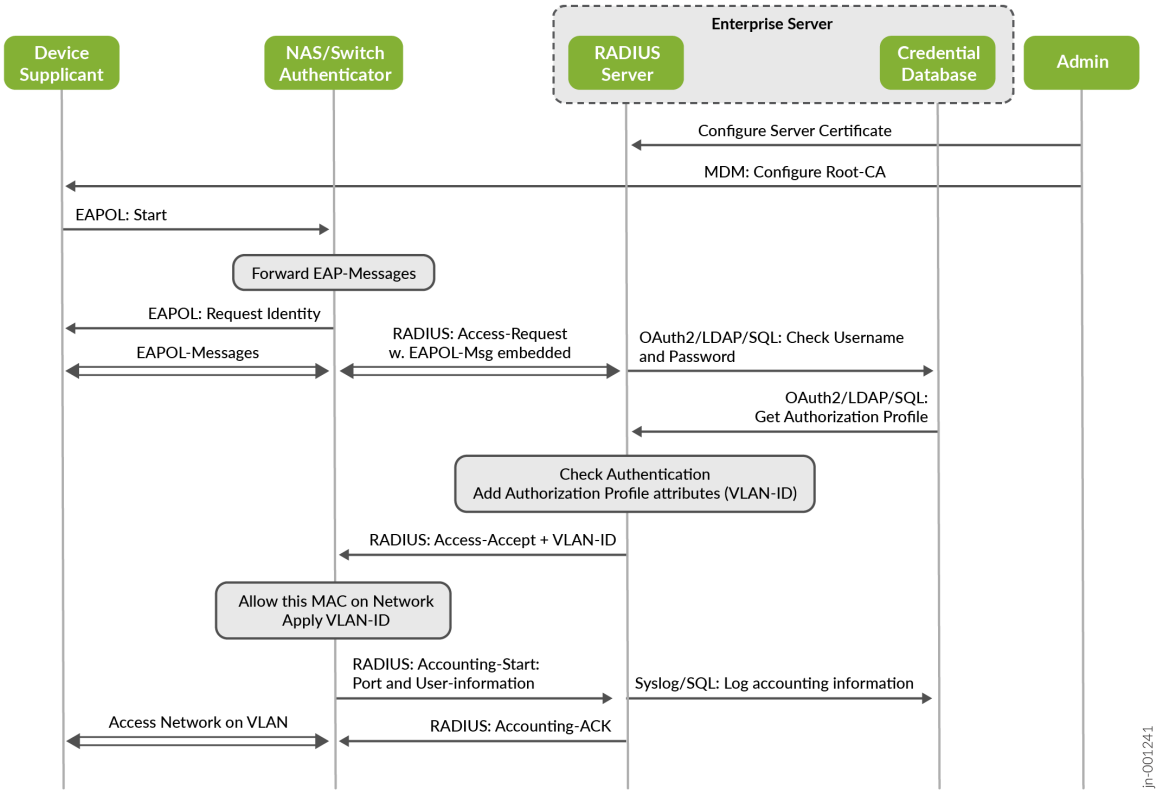


The example in Figure 13 on page 18 shows EAP-TTLS with an inner PAP authentication method which involves the following:

- An enterprise Public Key Infrastructure (PKI) is used for this EAP method to generate and pre-deploy the following certificate information:
  - The root CA, any intermediate signing CA, and the TLS server certificate (with public and private keys) on the RADIUS server.
  - The root CA and any intermediate signing CA via some enterprise device management method to the supplicant. This is needed to validate the certificate offered by the server.
- The supplicant then starts sending EAPoL messages that the authenticator forwards inside RADIUS messages to the authentication server to perform the following:
  - Build a secure TLS tunnel between the supplicant and the RADIUS server.
  - Inside the TLS tunnel (not shown in the below figure) username and password credentials are exchanged.
- Like the previous MAC address-based authentication, these credentials are exchanged against a credential database and verified on the authentication server.
- Should the credential check validate optional authorization, RADIUS AVPs may be added to the returned Access-Accept message. In the example, we enforce a certain VLAN to be used.

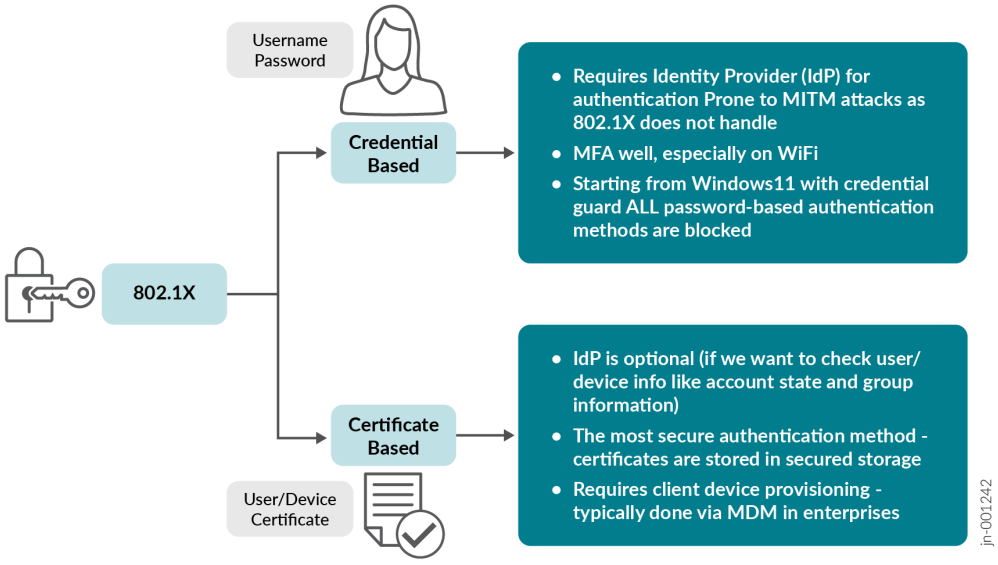
- The authenticator then oversees opening the access for the supplicant when it receives an Access-Accept message back from the authentication server and enforces all received authorization parameters (granting access to the client to a specific VLAN, in this example).

Figure 13: Example EAP-TTLS Authentication

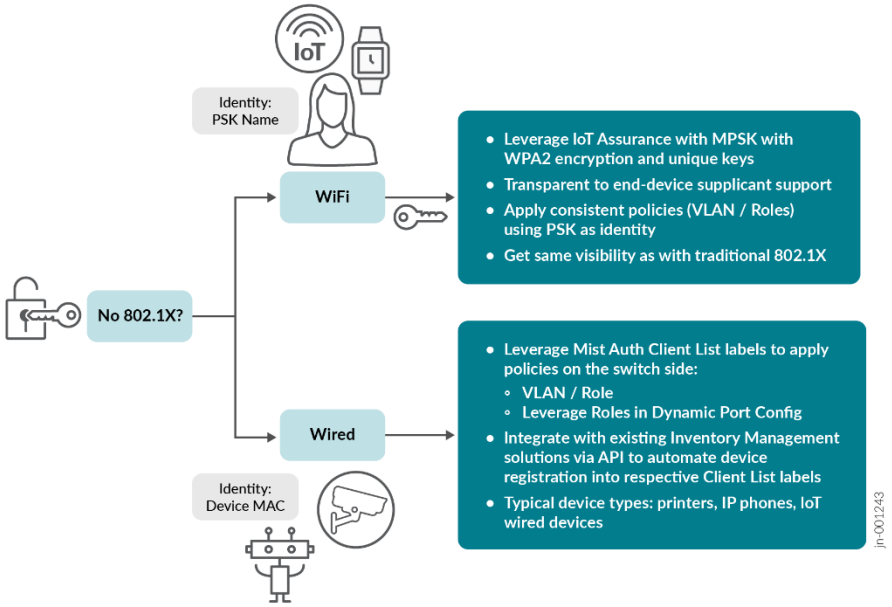


NOTE: For the purpose of better comparison with the previous example, we have removed the embedded dynamic cryptographic key material distribution and any later negotiation of an encrypted link between the supplicant and authenticator, typical for WPA2 and WPA3 enterprise WLAN clients.

In an enterprise network, you should use a strong EAP method where it is technically possible. Review the figure below to be able to select the best method.



Where it is technically not possible to use EAP, consider the options shown in the figure below.

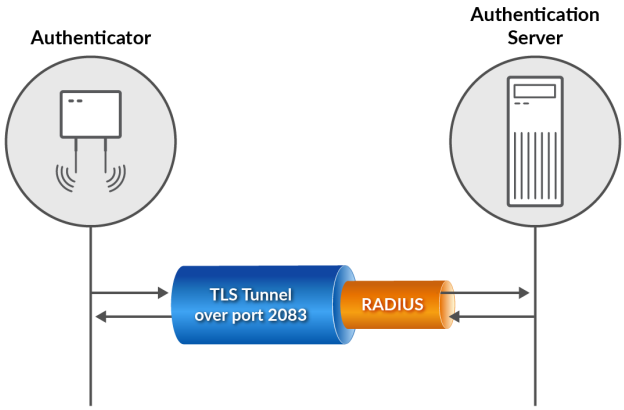


Architecture: Using a RadSec Tunnel Between the Authenticator and the RADIUS Server

The Juniper Mist Access Assurance solution uses an additional protocol/tunnel wrapped around the RADIUS UDP messages between the authenticator and the RADIUS server.

- This tunnel is based on the TLS V1.2 and V1.3 protocol, such as that used to protect a web server connection via HTTPS.
- The TCP destination port towards the RADIUS server is 2083.
- The RADIUS server, that is part of the Juniper Mist authentication cloud can be reached via the FQDN radsec.nac.mist.com.
- The Juniper Mist authentication cloud is responsible for creating a certificate and loading it on to the authenticator so that the TLS connection can be built with the authentication cloud. For this reason, Juniper Mist creates its own PKI for every organization.

Figure 14: Figure 14:RadSec Tunnel Usage



Attribute	Details
TLS Tunnel	Uses standard TLS 1.2/1.3
Port	TCP 2083
RadSec Authentication	Certificates on Authenticator and Authentication Server

RadSec Use Cases
Hotspot 2.0 authentication against SP
MistEdge RadSec Proxy
Cloud-hosted authentication
Eduroam type deployments with federation roaming

The reason for choosing this RadSec design has several technical reasons.

- It's a more robust transport when the RADIUS server is remote on the Internet.

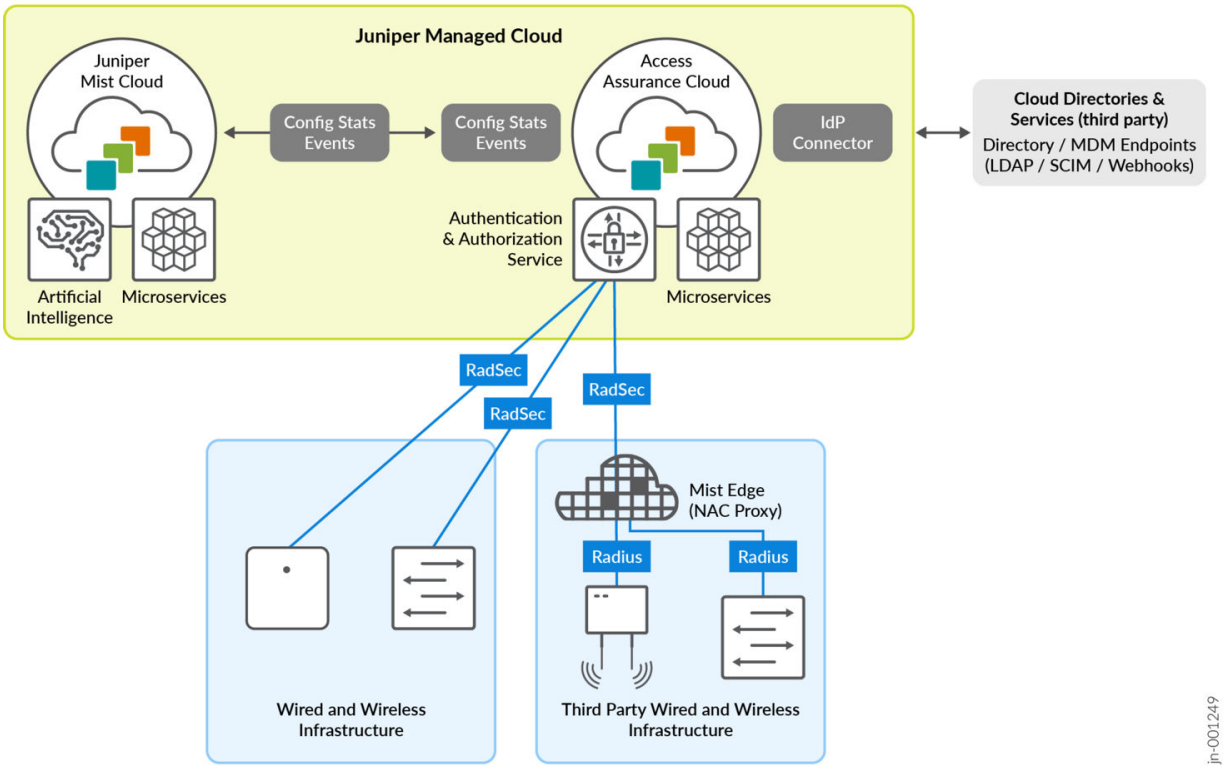
- Credentials and client information is secured from eavesdropping and man-in-the-middle attacks such as blastradius due to the RadSec TLS tunnel. This also relaxes the need for strong and individual shared-secrets defined between the RADIUS client and server.
- Enterprise firewall administrators are more confident when opening a TLS connection than opening remote UDP ports.
- The usage of a fixed FQDN enables the solution to select the nearest Juniper Mist authentication cloud via a global DNS load balancer. RadSec termination always happens to the nearest available Juniper Mist authentication cloud to optimize the latency.
- Should a third-party device or a Juniper switch not managed by the Juniper Mist cloud be used, then an additional proxy needs to be planned. This proxy acts like a replacement of the legacy on-site RADIUS and translates between RADIUS and RadSec towards the Juniper Mist authentication cloud. The Juniper Mist™ Edge appliance can be used to host this proxy service.
- Future usage of RADIUS Change of Authorization messages is very easy to embed as the Juniper Mist RADIUS server can use the existing TLS tunnel for those messages.

The important take away is that the legacy RADIUS server is no longer local and needs to be reachable via the corporate VPN. It's now a function of the Juniper Mist authentication cloud offer. Credential databases are also usually not local as customers start to embrace public IdPs such as Azure AD and Okta since nowadays not only client credentials need to be authenticated. The RadSec tunnel and configuration are managed by the Juniper Mist authentication cloud to enable this seamlessly.

Architecture: Juniper Mist Access Assurance Framework

The entire resulting framework for Juniper Mist Access Assurance is highlighted in the figure below. Note that the Access Assurance administrator uses and only sees the same Juniper Mist portal that he uses to manage switches and APs.

Figure 15: Overview Access Assurance Framework



With reference to the first authentication example shown in this chapter with a local RADIUS server and credential database (Figure 11 on page 15), compare it to Figure 16 on page 23. Note the differences in Figure 16 on page 23:

- The Juniper Mist authentication cloud oversees creating and loading the certificate onto the authenticator device for building the RadSec tunnel.
- The RADIUS server and credential database have been moved to the Juniper Mist authentication cloud.
- RadSec as TLS tunnel is now used between the authenticator and Juniper Mist Access Assurance (RADIUS server).
- All the remaining workflow is untouched and does not change the way it operates.

Figure 16: RadSec Implementation for MAB

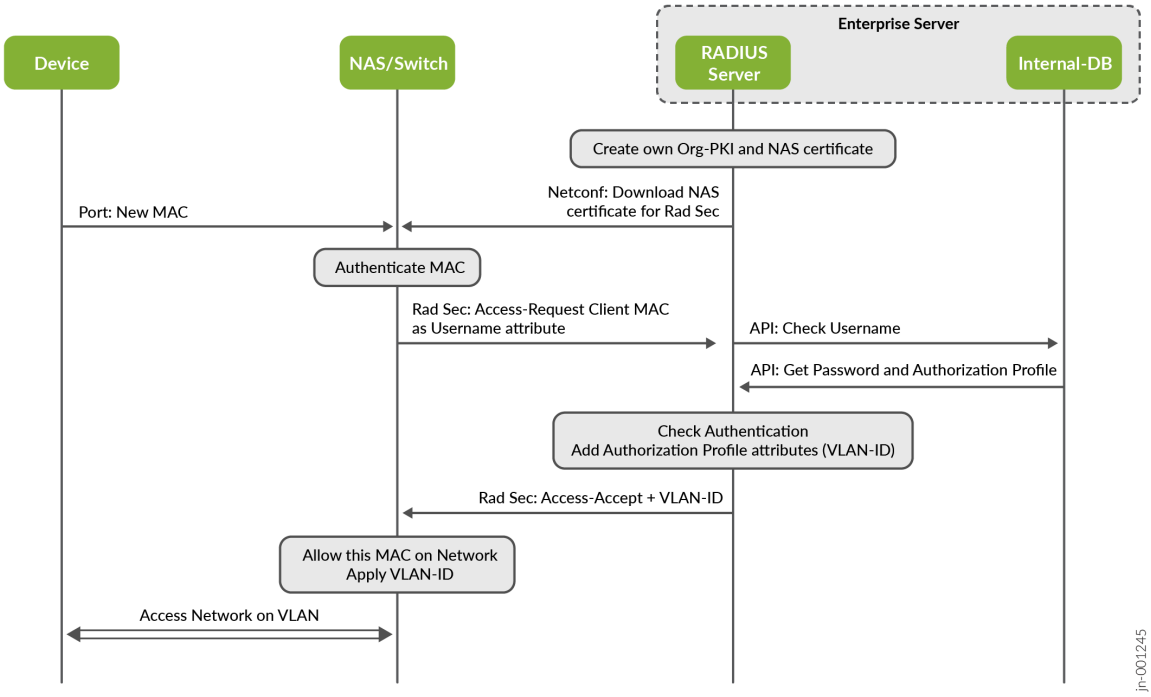


Figure 17 on page 24 adds a Juniper Mist Edge device as proxy for the above Figure 16 on page 23. This is in case you have third-party devices (switches and APs), or Juniper EX Switches not managed by the Juniper Mist cloud. If you have an existing RADIUS server, then you can use the Mist Edge proxy as a drop-in replacement without the need to change anything on the existing switch or AP configuration.

Figure 17: RadSec via Proxy and MAB

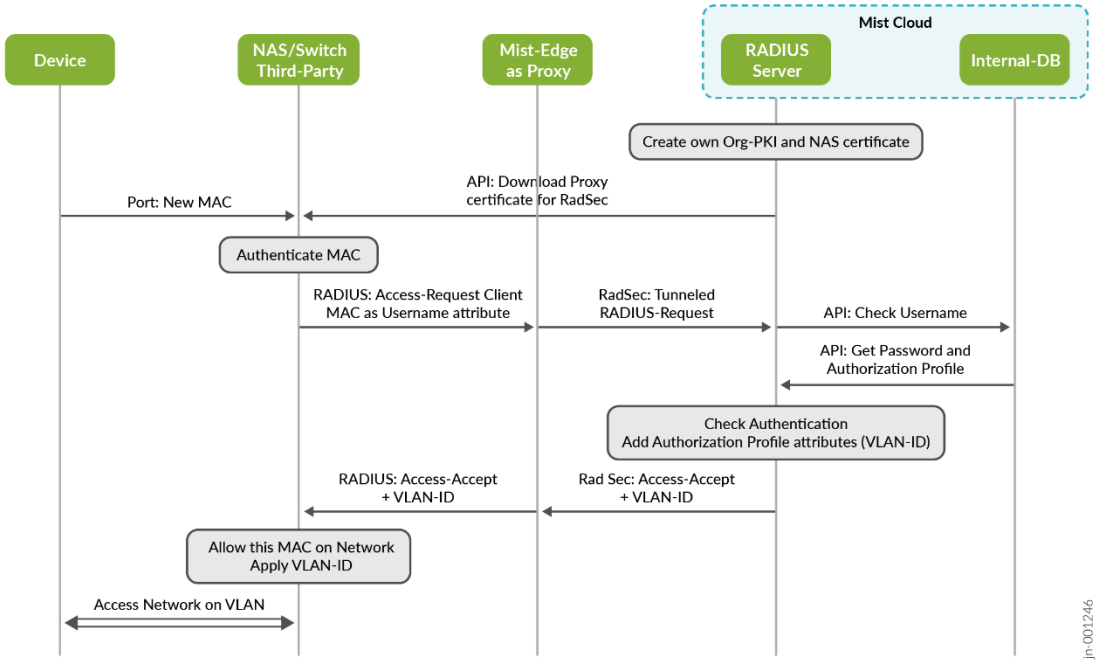
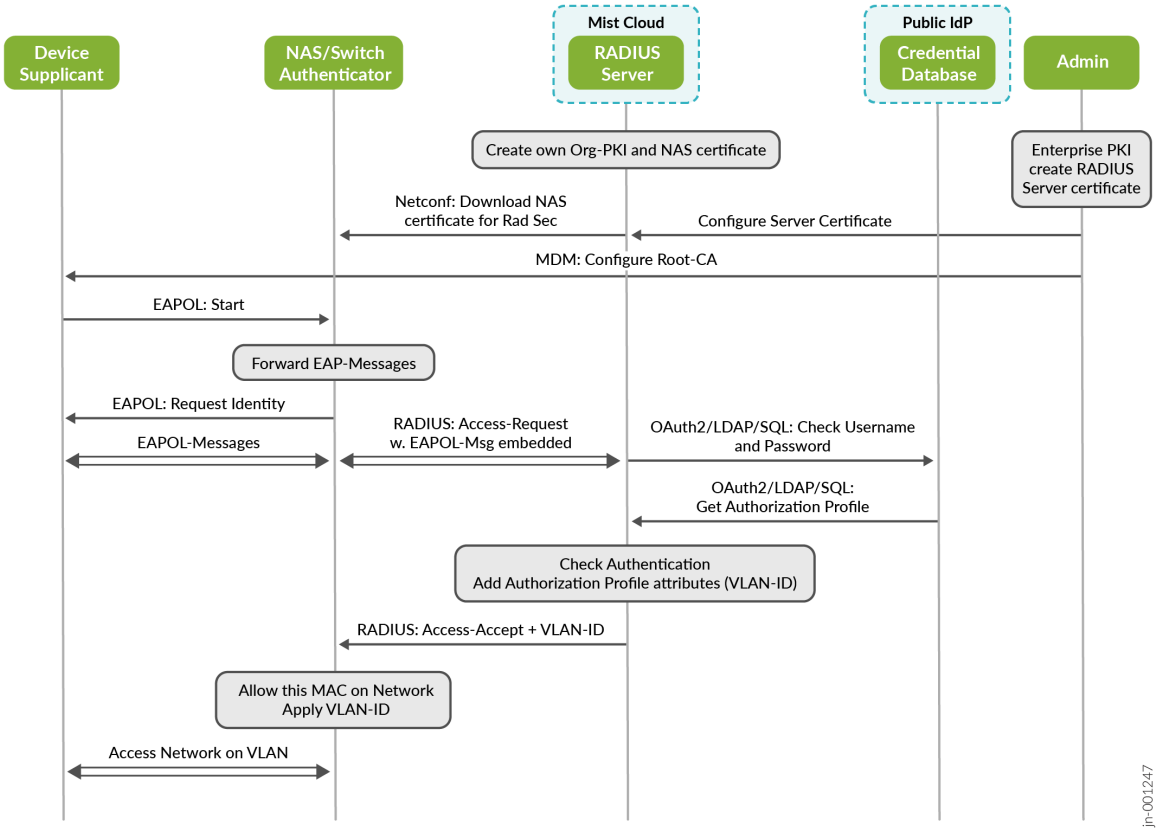


Figure 18 on page 25 shows what happens to EAP authentication methods when implementing RadSec and Juniper Mist Access Assurance:

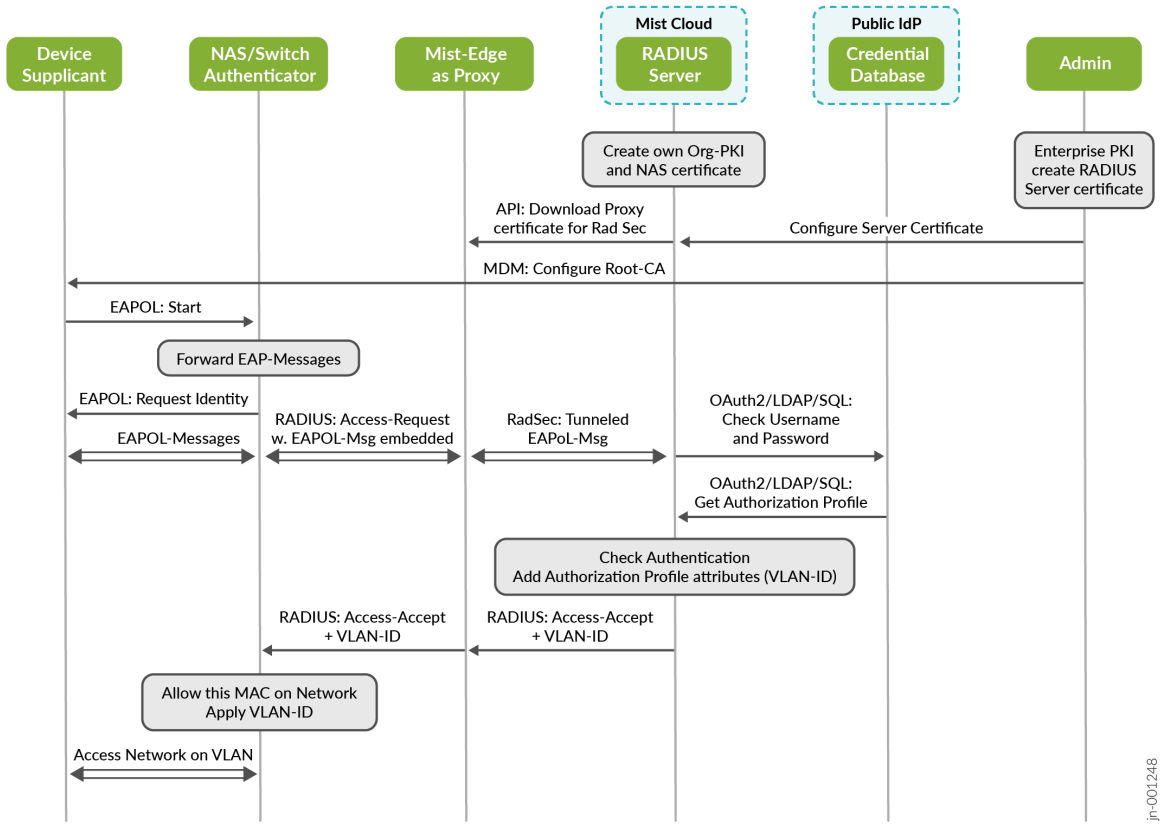
- The Juniper Mist authentication cloud oversees creating and loading the certificate onto the authenticator device for building the RadSec tunnel.
- The RADIUS server has been moved to the Juniper Mist authentication cloud.
- The credential database has been moved to a public IdP.
- RadSec as TLS tunnel is used now between the authenticator and Juniper Mist Access Assurance (RADIUS server).
- All the remaining workflow is untouched and does not change the way it operates.

Figure 18: RadSec with EAP Authentication and Public IdP



In Figure 19 on page 26, you can see the inclusion of a Juniper Mist Edge device as proxy for the above figure as previously shown.

Figure 19: RadSec via Proxy for EAP-Authentication with Public IdP



Implementation into Branch and Campus Fabric Designs

NAC solutions always take place where a client ingresses a network. You must be able to block traffic for unauthenticated devices or limit access via authorization enforcement at the lowest entry level of the network. By design, NAC solutions are independent from the transport that happens after network access. The NAC solution is not concerned with whether forwarding is based on VLANs, VXLAN, or routing. The table below lists the possible permutations:

Table 1: Access Assurance Implementation Details

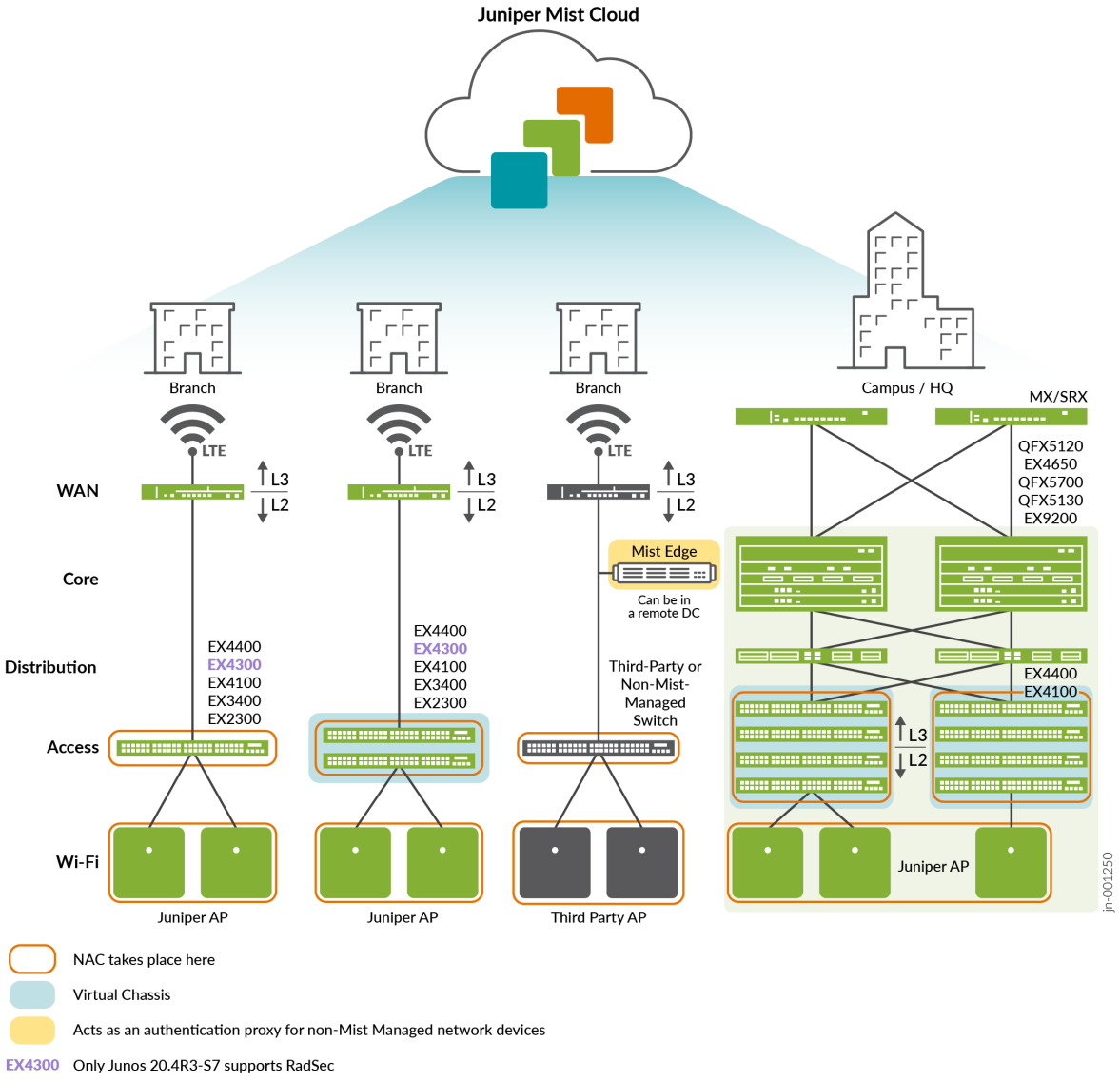
Location	Access method	Design	Mist Cloud managed?	NAC takes place at	RadSec supported?	Mist Edge proxy need?
Branch	Wired	Standalone Juniper Switch	Yes	Access Switch	Yes	No
Branch	Wired	Juniper Virtual Chassis	Yes	Access Switch	Yes	No
Branch	Wireless	Mist Access Point	Yes	Access Point	Yes	No
Branch	Wired	EOL Juniper Switch/VC	No	Access Switch	No	Yes
Branch	Wired	Juniper Switch/VC not Mist managed	No	Access Switch	No	Yes
Branch	Wired	Third-Party Switch	No	Access Switch	No	Yes
Branch	Wireless	Third-Party Access Point	No	Access Point	No	Yes
CF EVPN Multihoming / CRB / ERB	Wired	Standalone Juniper Switch	Yes	Access Switch / ToR	Yes	No
CF EVPN Multihoming / CRB / ERB	Wired	Juniper Virtual Chassis	Yes	Access Switch / ToR	Yes	No
CF EVPN Multihoming / CRB / ERB	Wireless	Mist Access Point	Yes	Access Point	Yes	No
CF EVPN Multihoming / CRB / ERB	Wired	EOL Juniper Switch/VC	No	Access Switch / ToR	No	Yes

Table 1: Access Assurance Implementation Details *(Continued)*

Location	Access method	Design	Mist Cloud managed?	NAC takes place at	RadSec supported?	Mist Edge proxy need?
CF EVPN Multihoming / CRB / ERB	Wired	Juniper Switch/VC not Mist managed	No	Access Switch / ToR	No	Yes
CF EVPN Multihoming / CRB / ERB	Wired	Third-Party Switch	No	Access Switch / ToR	No	Yes
CF EVPN Multihoming / CRB / ERB	Wireless	Third-Party Access Point	No	Access Point	No	Yes
CF IP-Clos	Wired	Standalone Juniper Switch	Yes	Access Switch / Leaf	Yes	No
CF IP-Clos	Wired	Juniper Virtual Chassis	Yes	Access Switch / Leaf	Yes	No
CF IP-Clos	Wireless	Mist Access Point	Yes	Access Point	Yes	No
CF IP-Clos	Wireless	Third-Party Access Point	No	Access Point	No	Yes

Figure 20 on page 29 describes the most common deployment options using Access Assurance.

Figure 20: Device Deployments with Access Assurance



Once the network access devices have been configured for RadSec (or RADIUS with Juniper Mist Edge proxy) the integration of the NAC solution according to customer requirements can begin. Here is an example of the typical tasks:

- Currently, the Juniper Mist Access Assurance solution supports the following authentication methods:
  - Password Authentication Protocol (PAP)—Mostly used for MAC address-based authentication.
  - Extensible Authentication Protocol-Transport Layer Security (EAP-TLS)—Mostly used for machine authentication as it needs certificates from an enterprise PKI pre-deployed to the supplicant.

- Extensible Authentication Protocol-Tunneled Transport Layer Security (EAP-TTLS) with PAP as inner authentication—Mostly used for user authentication when usernames and passwords need to be checked against an IdP.
- Less common: Protected Extensible Authentication Protocol-Transport Layer Security (PEAP-TLS) (note that PEAP-MS-CHAPv2 is NOT supported as backend methods such as OAuth2 do not support NTLM hashes).
- Less common: Tunnel Extensible Authentication Protocol (TEAP) with TLS. Consider moving to EAP-TLS.

Methods not on this list need to be discussed with the customer and how the devices using them can be changed. These methods are typically used by legacy devices and properly supporting them needs to be evaluated.

- Discuss which credential databases exist and how they can be integrated into the authentication and authorization process.
  - For MAC address-based authentication, Juniper Mist Access Assurance hosts a list of allowed MACs (you can also choose to identify a vendor by the OUI).
  - Public IdPs most commonly used include the following:
    - Azure AD
    - Okta leveraging OAuth2 between the Juniper Mist authentication cloud and the IdP.
    - Google Workspace leveraging LDAP over SSL (LDAPS) between the Juniper Mist authentication cloud and the IdP.
    - For other IdPs using OAuth2, contact your Juniper representative.
- Discuss the need for authorization profiles and how they can be applied to the network access device (especially when third-party vendor attributes are used).
- Discuss the need for an enterprise PKI:
  - Most customers have an existing PKI that can be used. Obtain the public root CA, any intermediate Cas, and ask the IT personnel to generate a public/private key for a TLS server in PEM format for the Juniper Mist Access Assurance RADIUS server.
  - The Juniper Mist PKI that is generated for each organization and used for the RadSec can be reused for EAP-TTLS when the supplicant is introduced to the Juniper Mist root CA.
  - All methods using a user or machine certificate (EAP-TLS, PEAP-TLS and TEAP-TLS) are expected to get certificates from the customer's enterprise PKI by the customer's mobile device management (MDM).

- Discuss the need for IoT devices as they usually do not leverage strong EAP authentication methods due to heavy battery usage for the computation required. The fallback is usually MAC-based authentication or MPSK for WLAN.
- Discuss the need for BYOD integration and portal pages.
- In educational environments, there may be a need for supporting roaming like Eduroam.
- Discuss the need of APs needing to act as a supplicant to be authenticated on a switch port. This is supported on non-EOL Juniper APs with newer firmware.

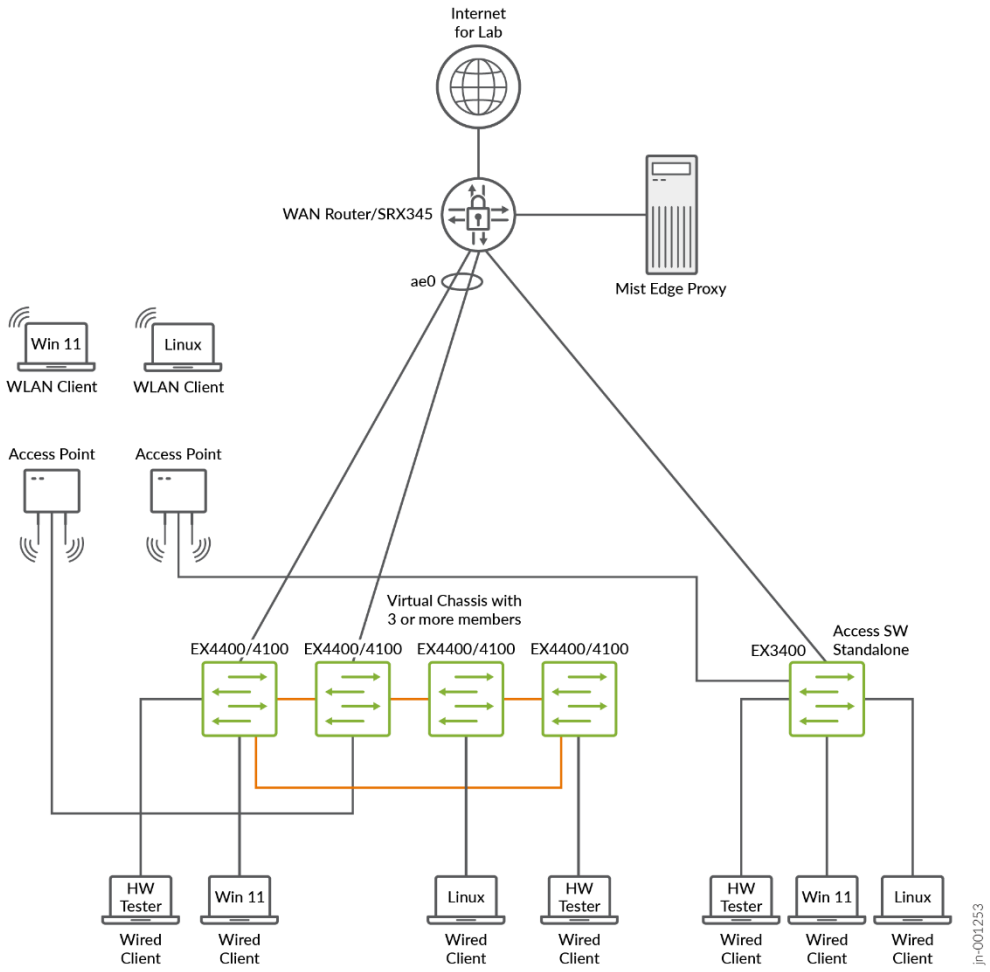
Validation Framework

IN THIS SECTION

Test Bed

For the evaluation of this JVD, a typical branch deployment was used. This provides repeatable scenarios using a minimum number of devices while still being able to execute all necessary functions (basically a single switch and AP are enough for most test cases). If you want to know more about such branch designs, Review the JVD Distributed Enterprise Branch EX.

Figure 21: Test Bed for this JVD



Platforms / Devices Under Test (DUT)

To review the software versions and platforms on which this JVD was validated by Juniper Networks, see the Validated Platforms and Software section in this document.

Test Bed Configuration

In the appendix section of this document, we share how some tests were performed. Contact Juniper Networks or your Juniper Networks account representative to obtain the full archive of the test bed configuration used for this JVD.

Test Objectives

IN THIS SECTION

Test Goals

The testing for this JVD was performed with the following goals in mind. Consult the Test Report of this JVD for more information.

- All Juniper Mist Access Assurance supported **authentication methods** need to be tested:
  - PAP. Used for MAB.
  - EAP-TLS.
  - EAP-TTLS with PAP using a public IdP.
  - PEAP-TLS (Limited to Win 11 client)
  - TEAP-TLS (Limited to Win 11 client)
- All major **authorization methods** need to be tested depending on what the network access device supports:
  - Wired switch authorization methods.
    - Dynamically assign a single access VLAN for a client.
    - Dynamically assign multiple VLAN as trunk and one native (for an attached AP).

        - Assign a pre-configured ACL filtered by filter-ID.
        - Session-timeout.
    - Wireless AP authorization methods.
        - Dynamically assign a single access VLAN for a client.
        - Dynamically assign a Mist Role for a client that then defines firewall filters.
- The **wired clients** for testing will be done using:
    - Spirent (mostly for MAB).
    - Windows 11 Professional.
    - Linux (Ubuntu 20.04 or else suggested).
- The **wireless clients** for testing will be done using:
    - Windows 11 Professional.
    - Linux (Ubuntu 20.04 or else suggested).
- There is a need to test the following combinations for network access devices:
    - Standalone Juniper access switch managed by Juniper Mist cloud.
    - Juniper Virtual Chassis with a minimum of 3 members managed by Juniper Mist cloud.
    - Juniper AP managed by Juniper Mist cloud.
    - Standalone unmanaged Juniper access switch with RADIUS using a Juniper Mist Edge proxy.
- Credential database integration testing (with and without group authorization profile assignment):
    - Mist Auth internal database (label-based) for MAB
    - Mist Auth internal database (endpoint page) for MAB
    - Azure AD for EAP-TTLS with PAP
    - Azure AD for EAP-TLS
    - Okta for EAP-TTLS with PAP
    - Okta for EAP-TLS
    - LDAPS for EAP-TTLS with PAP
- MDM integration:
    - Azure AD and Microsoft Intune integration with compliance authorization selection.

- Wi-Fi only topics:
  - Wi-Fi PSK authentication
  - Wi-Fi service-set identifier (SSID) authentication
  - Juniper AP as EAP-TLS supplicant
  - Wi-Fi roaming
  - WPA3 enterprise
  - Wi-Fi client onboarding with PSK portal
- Special topics
  - EAP-TLS client certificate attribute-based authorization selection
  - Switch CLI admin authentication (device-auth)

Test Non-Goals

- Testing with third-party switches and APs was archived through using Juniper Switches and AP in unmanaged mode with Mist-Edge as RadSec Proxy.
- Working with customer PKIs. We use either the automatic Mist PKI for each organization or a homegrown PKI.
- Testing Eduroam forwarding due to lab limitations.
- Testing dynamic assignment for group-based policies (GBP) as:
  - This is extensively tested by a JVD extension for IP Clos.
  - Testing with RADIUS AVP Juniper-Switching-Filter was performed and only the content of the string changes for dynamic GBP assignment.
- Testing with campus fabric. All JVDs for campus fabric already undergo rigorous testing with a third-party RADIUS server and MAB + EAP method testing to ensure authentication and authorization (VLAN/filter) assign. NAC solutions take care of the authentication and authorization when a new client accesses a network at the ingress access switch. They do not need to know how the transport is done after that point and are independent if you just forward via VLANs or VXLAN.
- Certificate expiration scenarios cannot be tested as we cannot change the clock setting on the Juniper Mist authentication cloud.

- Redundancy of the Juniper Mist Edge proxy device is not tested in this first phase. Keep in mind that the client does trigger the failover between RADIUS servers.
- Mobile Device Management testing exclusions:
  - Jamf is an MDM option for Apple supplicant was excluded due to lab limitations.
  - Testing with Airwatch was excluded due to lab limitations.

Recommendations

The following list of recommendations summarizes the best practices covered throughout this document:

- Most customers are expected to already have a PKI to leverage for any 802.1X EAP authentications.
  - To be able to use EAP-TTLS, customers' IT specialists must provide:
    - The public root CA and any intermediate signing CAs in PEM format. The Mist administrator must load those into **Organization > Certificates > Certificate Authorities**.
    - A public and private certificate for the RADIUS server (as a TLS-Server) in PEM format. The Mist administrator must load those into **Organization > Certificates > Certificate Authorities**.
    - It's expected that all customer supplicants have already learned the public root CA and any intermediate signing CA either manually or through MDM.
  - To be able to use EAP-TLS (or TEAP-TLS or PEAP-TLS) customers' IT specialists must provide:
    - The public root-CA and any intermediate signing CA in PEM format. The Mist administrator must load those into **Organization > Certificates > Certificate Authorities**.
    - A public and private certificate for the RADIUS server (as a TLS-Server) in PEM format. The Mist administrator must load those into **Organization > Certificates > Certificate Authorities**.
    - It's expected that all customer supplicants have already learned the public root CA and any intermediate signing CA either manually or through MDM.
    - A client certificate (signed by the PKI) for all customer supplicants. If that certificate is deployed manually or using an MDM is left up to the customer's preference.
- It's recommended to use the Mist PKI that is automatically generated for each organization for the purpose of:
  - Establishing the RadSec tunnels from all Juniper Mist-managed devices towards the authentication cloud.

    - EAP-TLS supplicant authentication of a Juniper AP towards the switch.
- When customers are using public IdPs, it's best to perform the integration with Juniper Mist cloud together with the customer's IT personnel for familiarity with the infrastructure and having the right authorization level.
- Authentication policies are executed similarly to a firewall. The evaluation execution happens top to bottom and wherever a match is found, it is executed and the policies below that one will not be evaluated. Hence, it is recommended that one positions the more specific (or exception) rules above the more generic rules.
- Use switch and WLAN templates for efficient configuration management. Configuration errors and unnecessary additional work can be avoided this way.
- When deciding how to manage switch port configurations dynamically:
    - Assigning VLANs and filters via RADIUS/NAC infrastructure is the recommended approach. Juniper Mist Access Assurance is designed to make this an easy task.
    - Using Dynamic Port Configuration is less preferred.
- Using MAC address-based authentication is only recommended when no EAP supplicant can be used on a client. MAC addresses are easy to fake by any attacker. Hence, it is suggested that those clients only get limited access to corporate resources.
- When using a Juniper Mist Campus Fabric perform the authentications via Out of Band management interfaces that are also used for Switch management towards Mist cloud. This will avoid potential issues with a too high overlay transport MTU towards a Mist-Edge as Proxy on standard MTU and EAP authentication with large certificates.

Appendix: Building the Topology and Testing Environment

IN THIS SECTION

In this appendix, we share information on how you can repeat the test cases that were executed for this JVD. The Figure 22 on page 40 configuration allows you to repeat all test cases for Access Assurance as long as you have the required minimal devices needed:

- A WAN router
  - Default gateway for all VLANs
  - DHCP server for all VLANs
  - Has at least one WAN connection towards the Internet where the Juniper Mist cloud and Juniper Mist authentication cloud is.
  - Has a VPN to a headquarters or uses local breakout of the traffic (as in our case).
  - Can define trunk ports with a native VLAN for inline management of attached switches and APs.
  - Optional: Trunk ports may use 802.3ad link aggregation with active LACP and the force-up option.
- A Standalone Switch
  - PoE support to power attached APs
  - Use a single uplink towards the WAN router
- Optional: A Virtual Chassis
  - Minimum two members for testing
  - PoE support to power attached APs
  - Use a LAG towards the WAN router
- Juniper APs
  - Powered using PoE from the switch.
  - Two or more at different switches to test roaming
- Optional: Juniper Mist Edge appliance
  - For this lab, directly attached to the local WAN router and not located remotely in a headquarters

- Mist authentication configuration for customer PKI
- Wired and Wireless clients to be tested
  - Make sure they support 801.1X EAP supplicants for the authentication method you want to test
  - Certificate management of these clients is not within the scope of Access Assurance. Customers can use an MDM or manual deployment

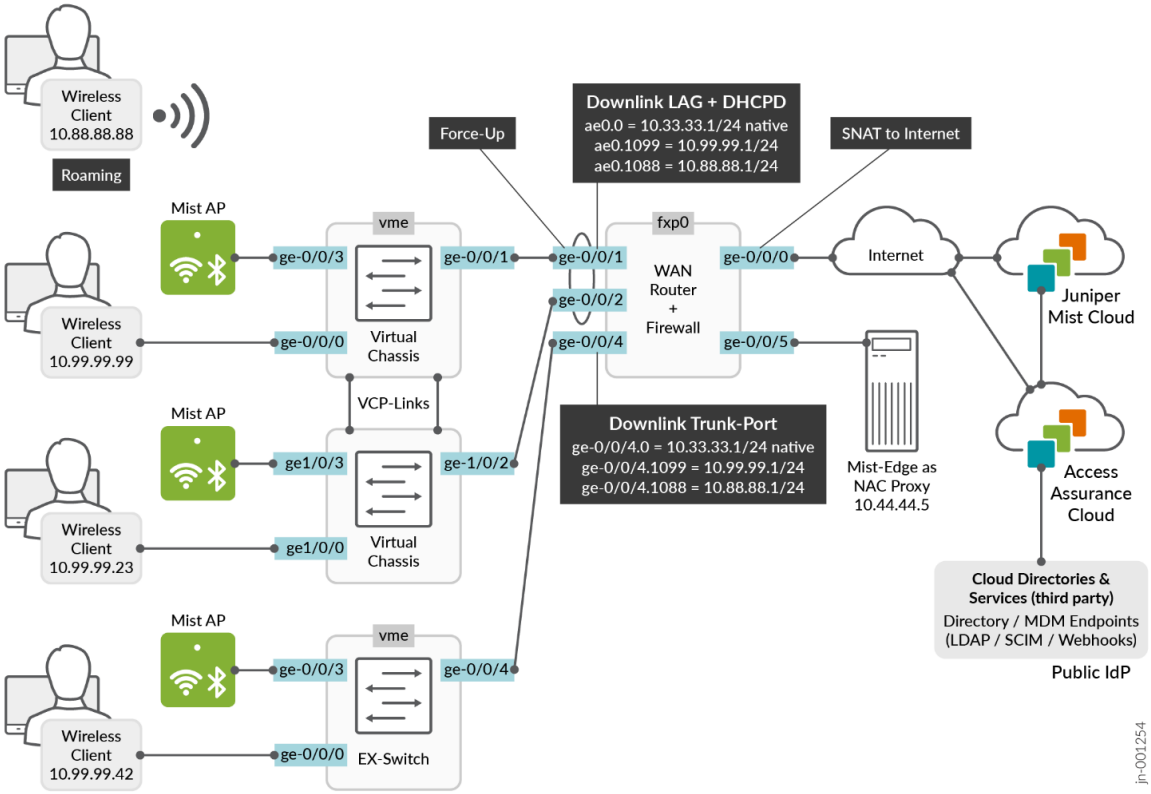
We are using four VLANs in this configuration for the minimally required functional design:

- VLAN1033 with the subnet range 10.33.33.0/24 is a native VLAN that is used for in-band management of the attached switches and APs.
- VLAN1099 with the subnet range 10.99.99.0/24 is used as a VLAN for attached wired clients.
- VLAN1088 with the subnet range 10.88.88.0/24 is used as a VLAN for attached WLAN clients.
- VLAN1044 with the subnet range 10.44.44.0/24 is used as a VLAN for the Juniper Mist Edge appliance.

All of these VLANs use the WAN router as the default gateway with the IP address 10.x.y.1 assigned to it, which also hands out DHCP lease to clients.

NOTE: VLAN1033 is getting treated as VLAN1 on the switch and AP. This is because the default VLAN on all Juniper switches is VLAN1 and is the access/native VLAN on all revenue ports.

Figure 22: Lab Topology Used in the Appendix



The recommended workflow for building such a lab is:

1. Deploy the WAN router and install
2. Deploy switches and Virtual Chassis and install
3. Deploy APs and install
4. OPTIONAL: Deploy Juniper Mist Edge and install
5. Juniper Mist authentication cloud certificate installation
6. Configure the client supplicants with certificates and the necessary EAP methods

WAN Router Installation and Configuration (Example for Branch Design)

In this chapter, we share configuration examples when using a Juniper Networks® SRX Series Firewalls as WAN router that is also managed by the Juniper Mist cloud in a simple branch design. Such a solution

is called a “full stack” solution as it enables you to manage all network devices located at a branch site within a single pane of glass.

If you have deployed a Juniper campus fabric, you can skip this chapter now as in most cases it does not apply to you. In most campus fabric designs, Layer 2 VLANs are terminated inside the fabric itself, and the WAN router oversees handling route forwarding Layer 3 information. The following JVD extension provides the information about WAN Router Integration into Campus Fabrics.

NOTE: Make sure the SRX device has an AppID license or else it cannot be managed by the Juniper Mist cloud. This is independent of whether you use it as a standalone firewall or as an SD-WAN router managing your VPN.

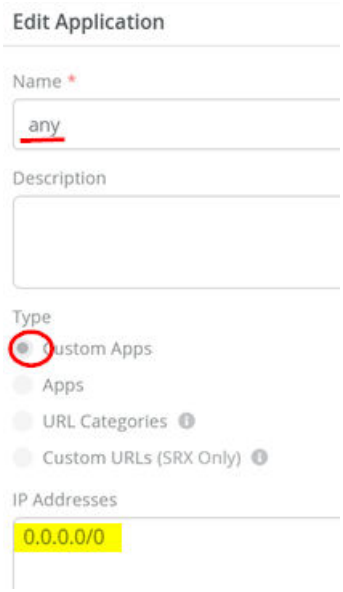
The following list of steps summarizes the process used to configure the WAN router in this chapter and is immediately followed by a detailed description of those steps:

- Define custom applications that present the destination IP ranges for internet and LAN segments.
- Define networks and VLANs
- Build a WAN Edge template describing:
  - WAN interfaces and their configuration
  - LAN interfaces and their configuration:
    - Default gateways for each network.
    - DHCP server settings for each network.
    - Binding of networks to interfaces along with possible LAG configurations.
  - Define traffic steering paths
  - Define application policies
  - Optional: Add additional Junos OS CLI commands
- Assign the template to sites
- Onboard a WAN Edge device and assign it to a site
- Check the configuration and status of the new WAN router

Go to **Organization > Applications** and check that there is an existing application with the following settings:

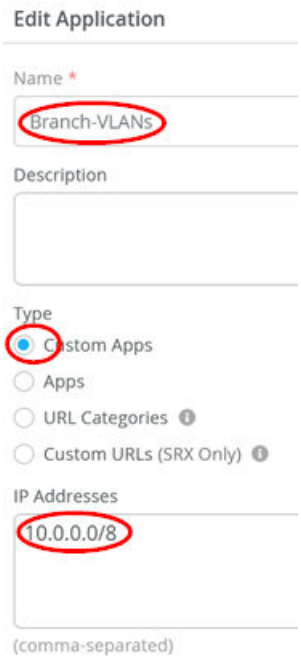
- Name=`any`

- Type=`Custom Apps`
- IP Addresses=`0.0.0.0/0`

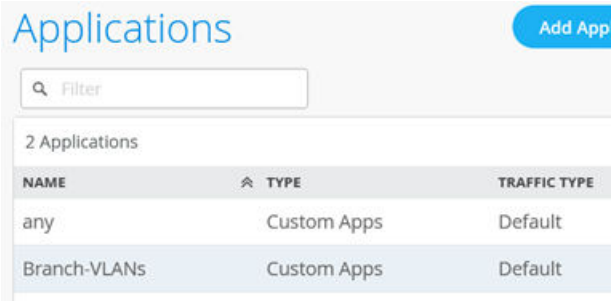


Add another application with the following settings:

- Name=`Branch-VLANs`
- Type=`Custom Apps`
- IP Addresses=`10.0.0.0/8`

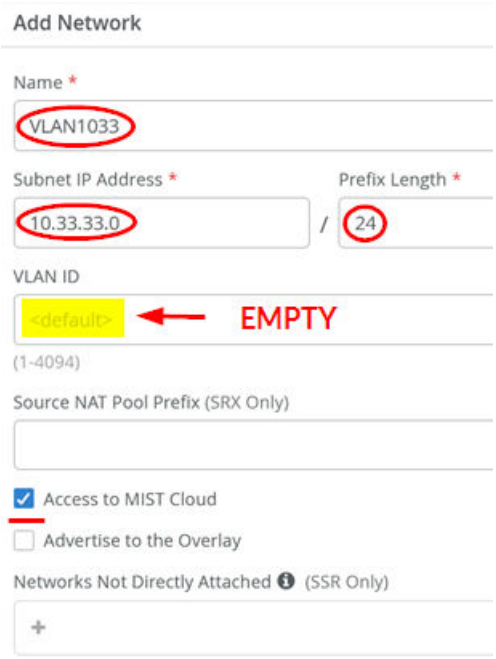


You should now see the two applications listed as shown below:



Go to **Organization > Networks** and add the first VLAN which is used to manage switches and APs:

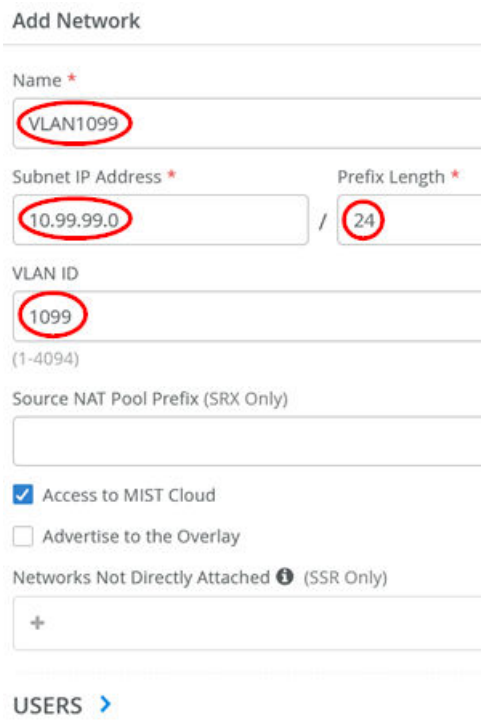
- Name=`VLAN1033`
- Subnet IP Address=`10.33.33.0`
- Prefix Length=`24`
- VLAN ID=`Leave this field empty`. (This is the native VLAN used for in-band management of the attached EX Series Switch as well as the AP).
- Access to Mist Cloud=`Enabled`. (This must be enabled for the attached switches and AP to be managed by the Juniper Mist cloud).



Then, add the second VLAN (in our topology we use this for wired clients):

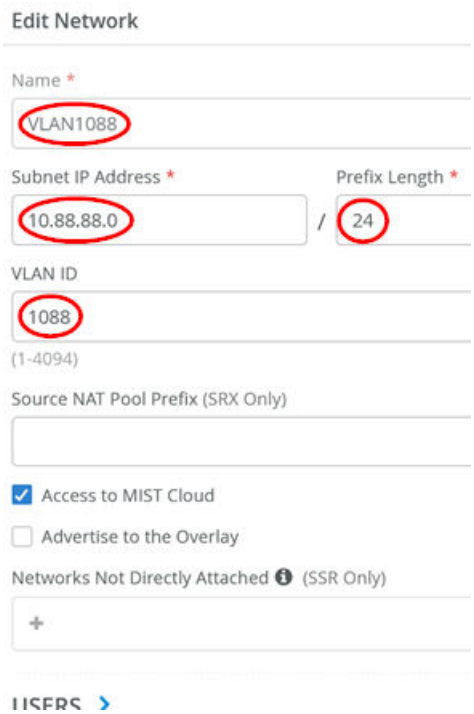
- Name=`VLAN1099`

- Subnet IP Address=`10.99.99.0`
- Prefix Length=`24`
- VLAN ID=`1099`



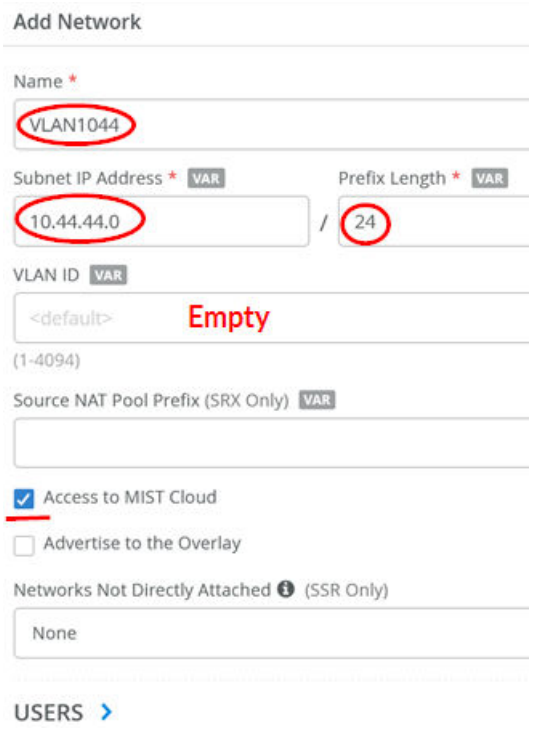
Then, add the third VLAN (in our topology, we use it for wireless clients attached to APs)

- Name=`VLAN1088`
- Subnet IP Address=`10.88.88.0`
- Prefix Length=`24`
- VLAN ID=`1088`

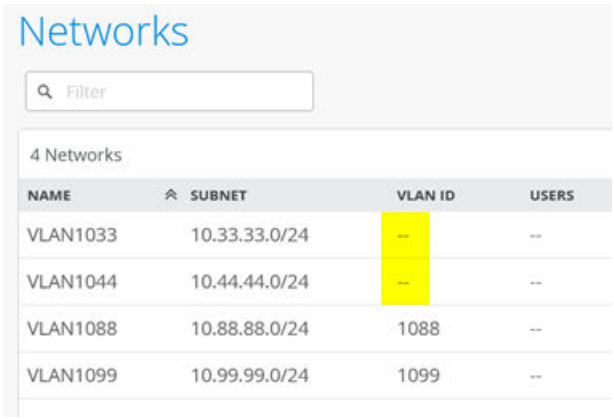


Add the last VLAN (in our topology, we use it for accessing the Juniper Mist Edge device's out-of-band management port)

- Name=`VLAN1044`
- Subnet IP Address=`10.44.44.0`
- Prefix Length=`24`
- VLAN ID=Leave this field empty. This is the native VLAN used towards the Juniper Mist Edge.
- Access to Juniper Mist cloud=`Enabled`. This must be enabled for the attached switches and AP to be managed by the Juniper Mist cloud.



Review the four networks and verify that no VLAN ID is set for the switch and AP management network and the Juniper Mist Edge attach, since this is a native VLAN on the downlink trunk.



The following JSON template may be used to configure the branch WAN router. Alternatively, manual configuration steps for the branch WAN router are listed immediately after the JSON template.

```
{
  "type": "standalone",
  "port_config": {
    "ge-0/0/0": {
      "usage": "wan",
      "name": "wan",
```

```
      "ip_config": {
        "type": "dhcp"
      }
    },
    "ge-0/0/15": {
      "usage": "wan",
      "name": "wan2",
      "ip_config": {
        "type": "dhcp"
      }
    },
    "cl-1/0/0": {
      "usage": "wan",
      "name": "lte",
      "wan_type": "lte",
      "ip_config": {
        "type": "dhcp"
      }
    },
    "ge-0/0/1-2": {
      "networks": [
        "VLAN1033",
        "VLAN1099",
        "VLAN1088"
      ],
      "usage": "lan",
      "aggregated": true,
      "ae_disable_lacp": false,
      "ae_lacp_force_up": true,
      "ae_idx": 0,
      "redundant": false,
      "critical": false,
      "disabled": false
    },
    "ge-0/0/4": {
      "networks": [
        "VLAN1033",
        "VLAN1088",
        "VLAN1099"
      ],
      "usage": "lan",
      "aggregated": false,
      "redundant": false,
```

```
      "critical": false,
      "disabled": false
    },
    "ge-0/0/5": {
      "networks": [
        "VLAN1044"
      ],
      "usage": "lan",
      "aggregated": false,
      "redundant": false,
      "critical": false,
      "disabled": false
    }
  },
  "ip_configs": {
    "VLAN1033": {
      "type": "static",
      "ip": "10.33.33.1",
      "netmask": "/24"
    },
    "VLAN1099": {
      "type": "static",
      "ip": "10.99.99.1",
      "netmask": "/24"
    },
    "VLAN1088": {
      "type": "static",
      "ip": "10.88.88.1",
      "netmask": "/24"
    },
    "VLAN1044": {
      "type": "static",
      "ip": "10.44.44.1",
      "netmask": "/24"
    }
  },
  "dhcpd_config": {
    "enabled": true,
    "VLAN1033": {
      "type": "local",
      "ip_start": "10.33.33.10",
      "ip_end": "10.33.33.250",
      "gateway": "10.33.33.1",
```

```
      "dns_servers": [
        "8.8.8.8",
        "9.9.9.9"
      ],
      "options": {}
    },
    "VLAN1099": {
      "type": "local",
      "ip_start": "10.99.99.10",
      "ip_end": "10.99.99.250",
      "gateway": "10.99.99.1",
      "dns_servers": [
        "8.8.8.8",
        "9.9.9.9"
      ],
      "options": {}
    },
    "VLAN1088": {
      "type": "local",
      "ip_start": "10.88.88.10",
      "ip_end": "10.88.88.250",
      "gateway": "10.88.88.1",
      "dns_servers": [
        "8.8.8.8",
        "9.9.9.9"
      ],
      "options": {}
    },
    "VLAN1044": {
      "type": "local",
      "ip_start": "10.44.44.10",
      "ip_end": "10.44.44.250",
      "gateway": "10.44.44.1",
      "dns_servers": [
        "8.8.8.8",
        "9.9.9.9"
      ],
      "options": {},
      "lease_time": 86400,
      "fixed_bindings": {}
    }
  },
  "path_preferences": {
```

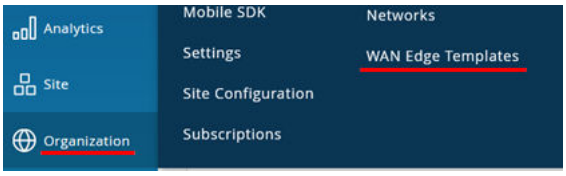
```
    "wan": {
      "paths": [
        {
          "type": "wan",
          "name": "wan"
        }
      ]
    },
    "LAN": {
      "strategy": "ordered",
      "paths": [
        {
          "type": "local",
          "networks": [
            "VLAN1033"
          ]
        },
        {
          "type": "local",
          "networks": [
            "VLAN1099"
          ]
        },
        {
          "type": "local",
          "networks": [
            "VLAN1088"
          ]
        },
        {
          "type": "local",
          "networks": [
            "VLAN1044"
          ]
        }
      ]
    }
  },
  "service_policies": [
    {
      "name": "inside_Branch_hairpin",
      "tenants": [
        "VLAN1033",
```

```
        "VLAN1088",
        "VLAN1099",
        "VLAN1044"
      ],
      "services": [
        "Branch-VLANs"
      ],
      "action": "allow",
      "path_preference": "LAN",
      "idp": {
        "enabled": false
      }
    },
    {
      "name": "Internet",
      "tenants": [
        "VLAN1033",
        "VLAN1099",
        "VLAN1088",
        "VLAN1044"
      ],
      "services": [
        "any"
      ],
      "action": "allow",
      "path_preference": "wan",
      "idp": {
        "enabled": false
      }
    }
  ],
  "bgp_config": {},
  "routing_policies": {},
  "extra_routes": {},
  "vrf_instances": {},
  "tunnel_configs": {},
  "oob_ip_config": {
    "type": "dhcp",
    "node1": {
      "type": "dhcp"
    }
  },
  "ntp_servers": [
```

```
    "time.google.com"
  ],
  "dns_servers": [
    "8.8.8.8",
    "9.9.9.9"
  ],
  "tunnel_provider_options": {
    "jse": {},
    "zscaler": {}
  },
  "additional_config_cmds": [
    "set security zones security-zone VLAN1033 host-inbound-traffic system-services ping",
    "set security zones security-zone VLAN1044 host-inbound-traffic system-services ping",
    "set security zones security-zone VLAN1099 host-inbound-traffic system-services ping",
    "set security zones security-zone VLAN1088 host-inbound-traffic system-services ping"
  ],
  "ospf_areas": {},
  "ospf_config": {
    "enabled": false,
    "areas": {}
  },
  "name": "Branch-WAN-Router"
}
```

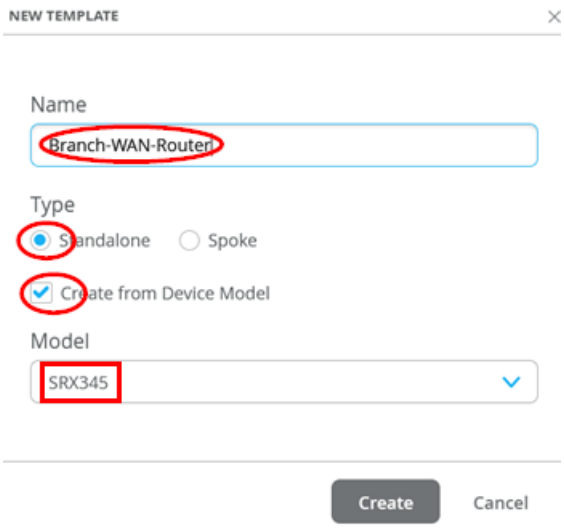
When not using the JSON template, execute the following steps instead to configure the branch WAN router:

Go to **Organization > WAN Edge Templates**:



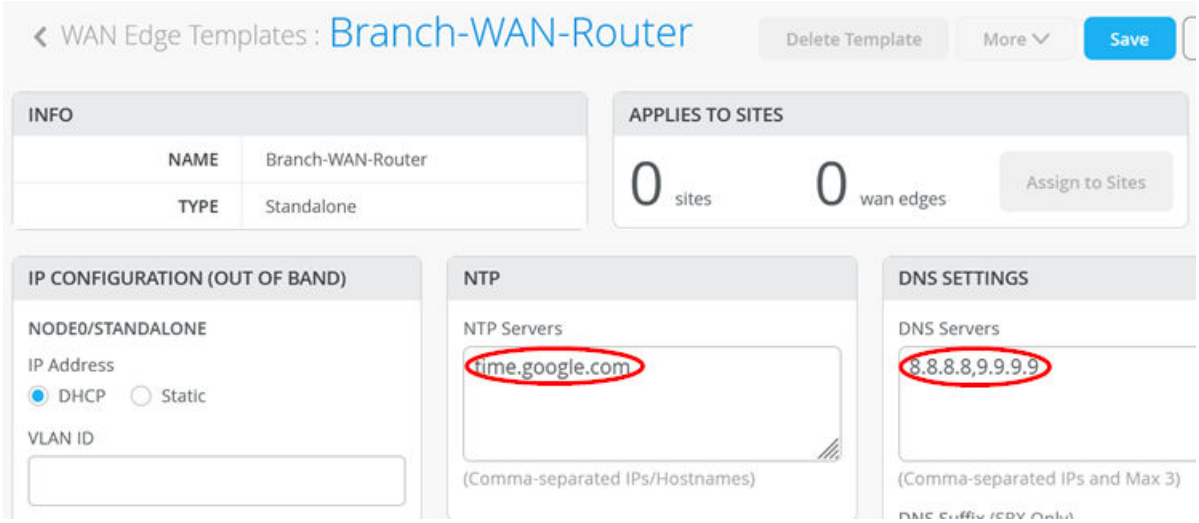
Create a new template with the following parameters:

- Name=`Branch-WAN-Router`
- Type=`Standalone`
- Create from Device Model=`Checked`
- Model=`<Select your Model>`

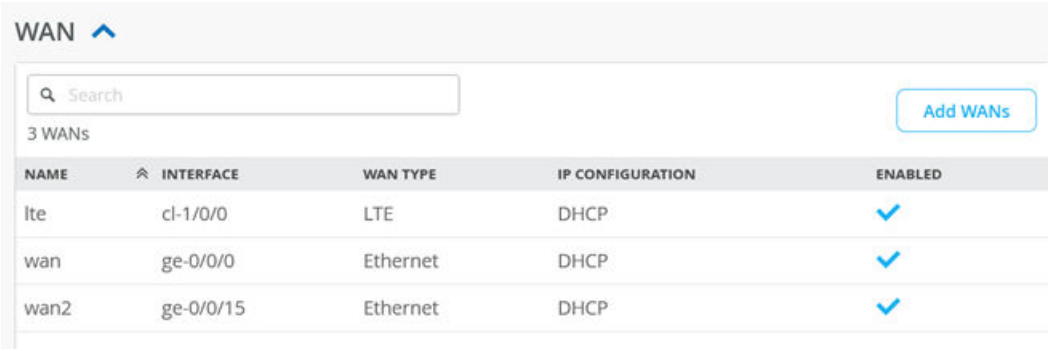


After the template has been created, start with basic configuration settings based on your environment such as the following:

- NTP=`time.google.com`
- DNS Servers=`8.8.8.8, 9.9.9.9`

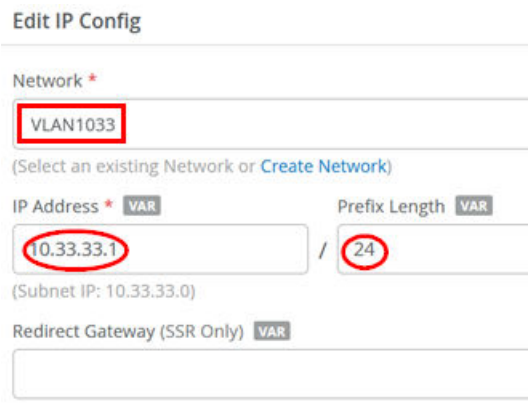


When you check the template, you should see the following preconfigured WAN interfaces. We are going to use the “wan” ge-0/0/0 interface to obtain a DHCP lease from the broadband router.



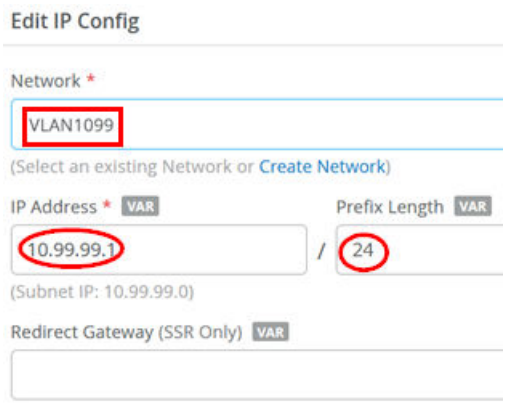
We are going to modify the LAN interfaces of this template. Delete the preconfigured “lan” interface (not shown here). Then, create a first IP configuration:

- Network=`VLAN1033`
- IP Address=`10.33.33.1`
- IP-Prefix Length=`24`



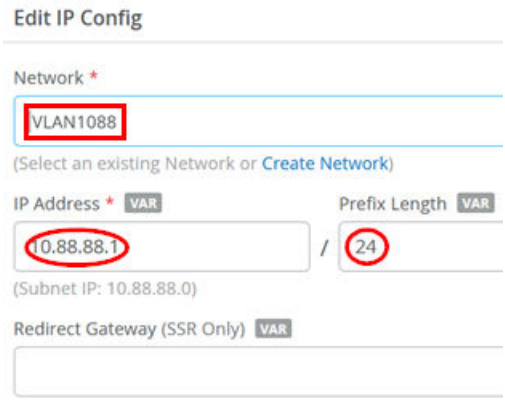
The second IP configuration is:

- Network=`VLAN1099`
- IP Address=`10.99.99.1`
- IP-Prefix Length=`24`



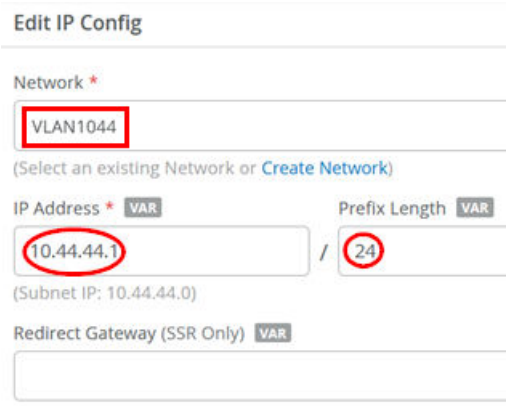
The third IP configuration is:

- Network=`VLAN1088`
- IP Address=`10.88.88.1`
- IP-Prefix Length=`24`



The last IP configuration is:

- Network=`VLAN1044`
- IP Address=`10.44.44.1`
- IP-Prefix Length=`24`



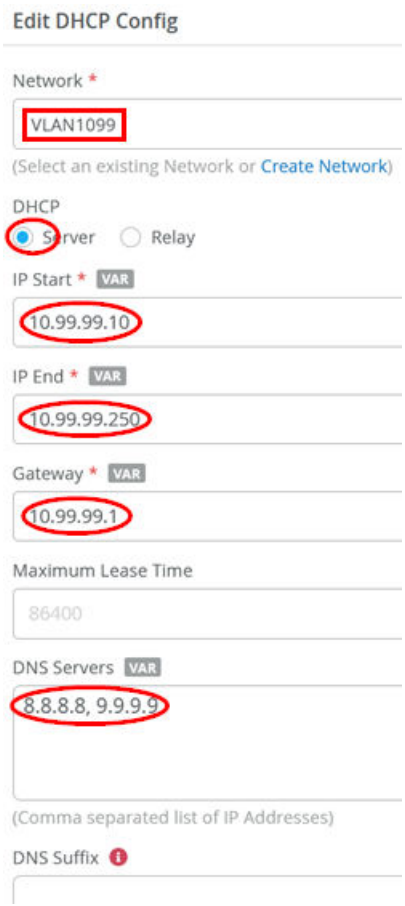
The resulting IP configuration should now look like the figure below:

Now, add the first DHCP server configuration:

- Network=`VLAN1033`
- DHCP=`Server`
- IP Start=`10.33.33.10`
- IP End=`10.33.33.250`
- Gateway=`10.33.33.1`
- DNS Servers=`8.8.8.8, 9.9.9.9`

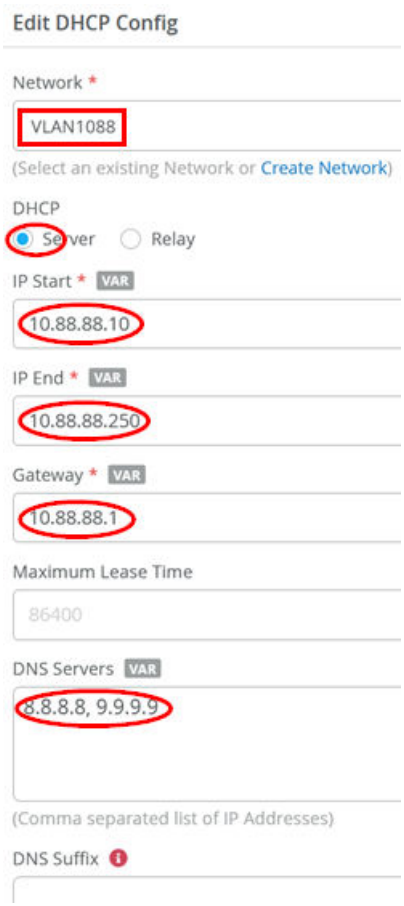
Then, add the second DHCP server configuration:

- Network=`VLAN1099`
- DHCP=`Server`
- IP Start=`10.99.99.10`
- IP End=`10.99.99.250`
- Gateway=`10.99.99.1`
- DNS Servers=`8.8.8.8, 9.9.9.9`



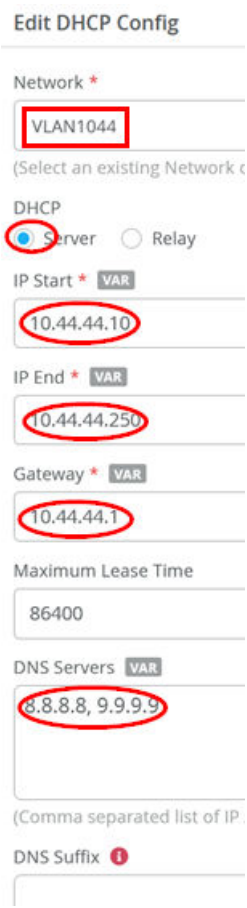
Then add the third DHCP server configuration:

- Network=`VLAN1088`
- DHCP=`Server`
- IP Start=`10.88.88.10`
- IP End=`10.88.88.250`
- Gateway=`10.88.88.1`
- DNS Servers=`8.8.8.8, 9.9.9.9`

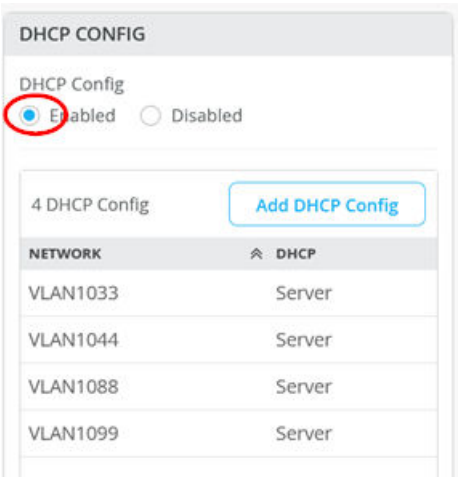


Then, add the last DHCP server configuration:

- Network=`VLAN1044`
- DHCP=`Server`
- IP Start=`10.44.44.10`
- IP End=`10.44.44.250`
- Gateway=`10.44.44.1`
- DNS Servers=`8.8.8.8, 9.9.9.9`



The resulting DHCP server configuration should look like the figure below:

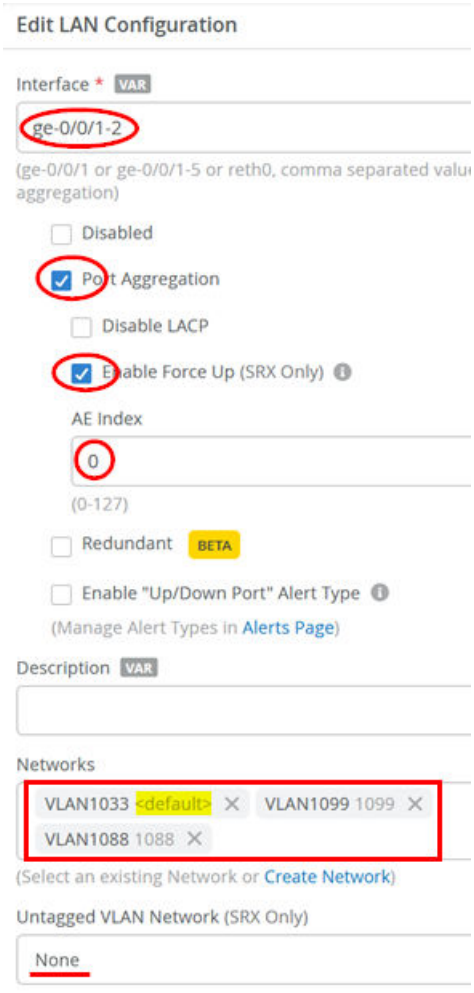


Next, build the LAN interface configurations. The first LAN interface is a LAG with force-up option towards the Virtual Chassis:

- Interface=`ge-0/0/1-2`

- Port Aggregation=`Enabled`
- Enable Force-Up=`Enabled`
- AE Index=`0`
- Networks= `VLAN1033, VLAN1099, VLAN1088`
- Untagged VLAN Network=`None` (as VLAN1033 is without any tag we do not need to tweak this)

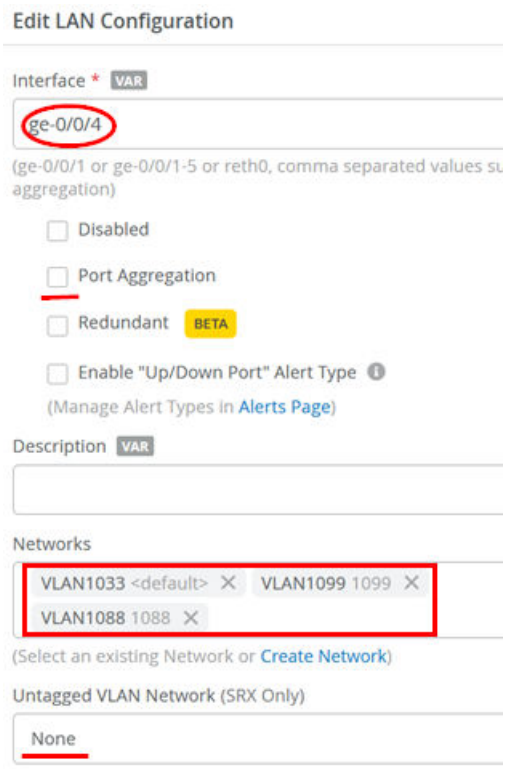
Figure 23: LAG with force-up



The second LAN interface is a normal trunk port for a single attached switch:

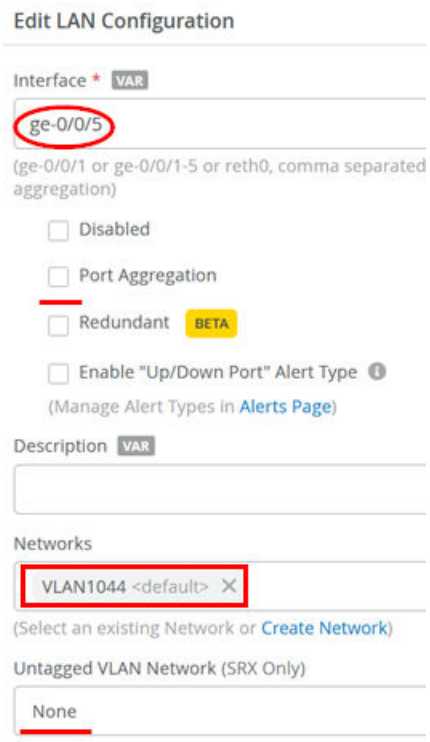
- Interface=`ge-0/0/4`
- Port Aggregation=`Unchecked`

- Networks= `VLAN1033, VLAN1099, VLAN1088`
- Untagged VLAN Network=`None` (as VLAN1033 is without any tag we do not need to tweak this)

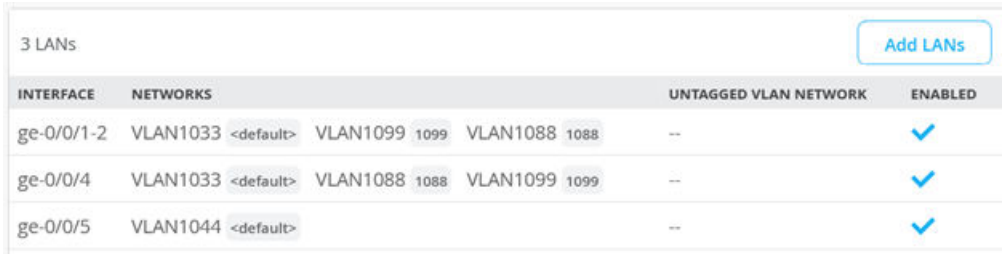


The third LAN interface is an access port with a single VLAN configured to integrate the Juniper Mist Edge into the management network:

- Interface=`ge-0/0/5`
- Port Aggregation=`Unchecked`
- Networks= `VLAN1044`
- Untagged VLAN Network=`None` (since VLAN1044 is untagged, we do not need to tweak this)

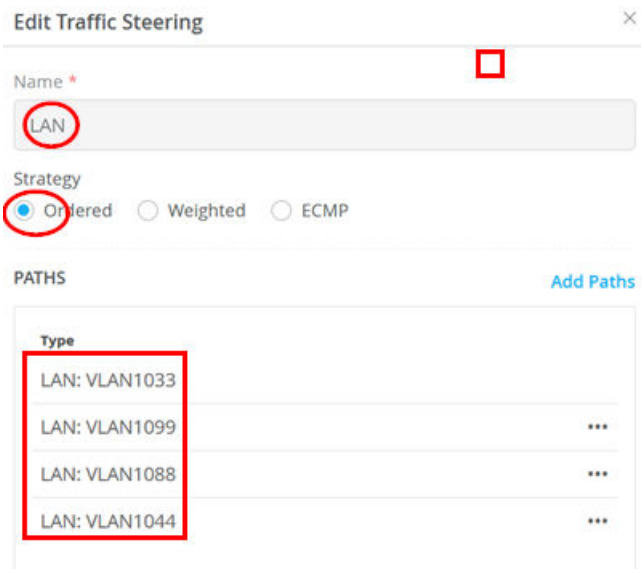


The resulting LAN interface configuration should now look like the figure below:

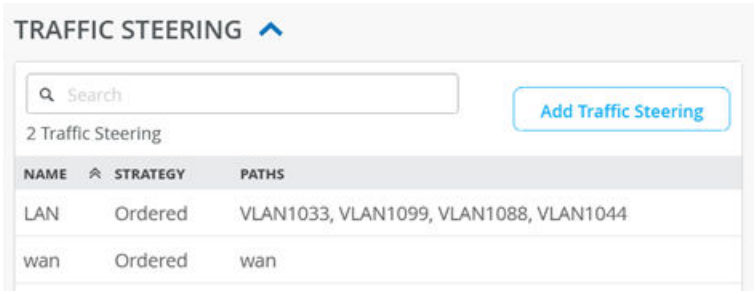


The next step is adding a new destination to the traffic steering policy. This is used to enable communication between the local VLANs, which is required in our example. Add a new traffic steering policy using the following settings:

- Name=`LAN`
- Strategy=`Ordered`
- Paths:
  - Type=`LAN: VLAN1033`
  - Type=`LAN: VLAN1099`
  - Type=`LAN: VLAN1088`
  - Type=`LAN: VLAN1044`



You should now see the following for the traffic steering destinations:

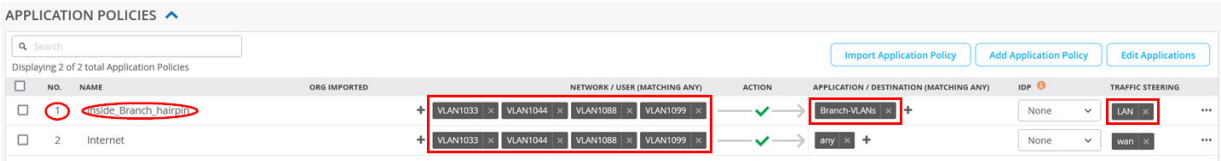


Implement Table 2 on page 64 for application policies. Parts should already exist that you only need to modify.

Table 2: Application Policies

Serial Number	Rule Name	Network	Action	Destination	Steering
1	Inside_Branch_hairpin	VLAN1033, VLAN1044, VLAN1088, VLAN1099	Pass	Branch-VLANs	LAN
2	Internet	VLAN1033, VLAN1044, VLAN1088, VLAN1099	Pass	any	wan

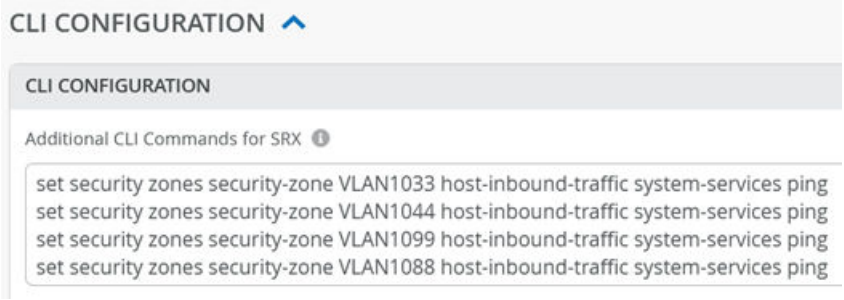
You should now see the following configuration for application policies after implementing the above table:



In the current version, clients on the LAN side cannot get an answer when sending ICMP pings to the WAN router as their local gateway. However, receiving pings is crucial for any local debugging. Hence, it is highly recommended that you add some additional Junos OS CLI commands to enable pings for any wired or wireless clients towards the WAN router as the local gateway of the VLAN they are attached to. See the example below:

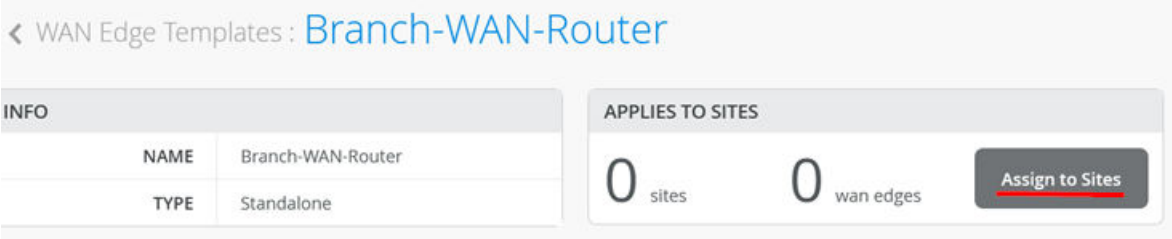
```
set security zones security-zone VLAN1033 host-inbound-traffic system-services ping
set security zones security-zone VLAN1044 host-inbound-traffic system-services ping
set security zones security-zone VLAN1099 host-inbound-traffic system-services ping
set security zones security-zone VLAN1088 host-inbound-traffic system-services ping
```

In the portal, it should look like the figure below:

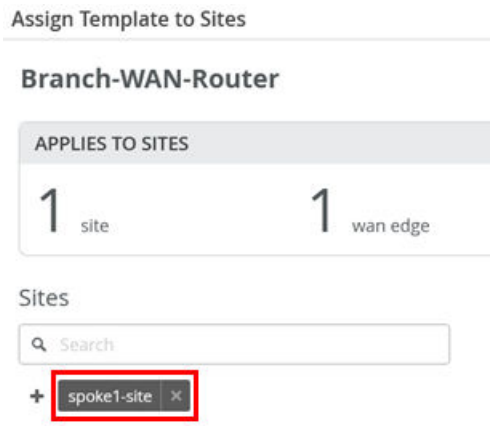


Click **Save** to save the template now.

You must assign a site for this template or else it won't be used on any device.



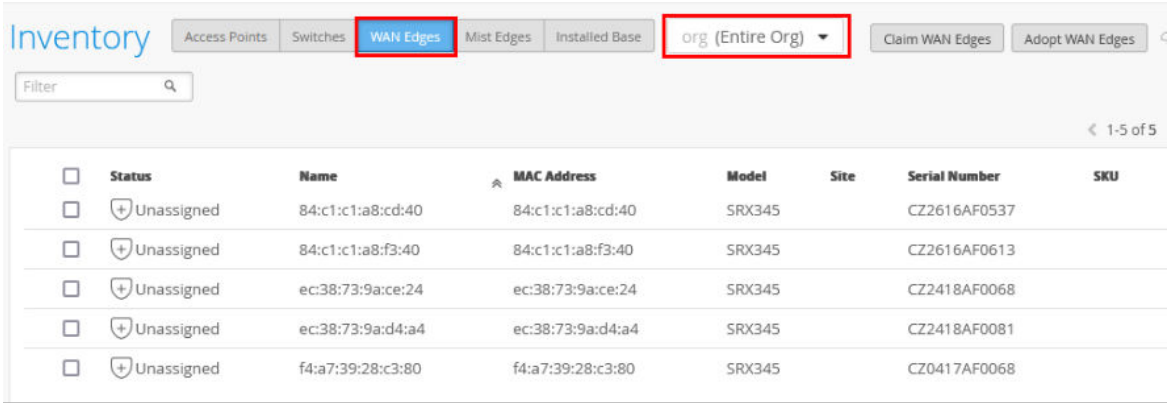
Here, we add the Spoke1 site to the template, which is where our switches are located.



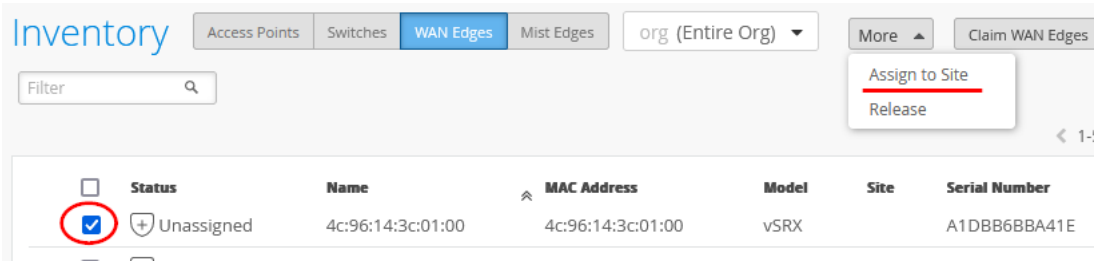
To assign your SRX Series Firewalls to sites, the devices must be present in the Juniper Mist inventory. You can claim or adopt your SRX Series Firewalls to onboard it into the Juniper Mist cloud. After the device is onboarded, the organization inventory shows the device.

To assign an SRX Series Firewall to a site:

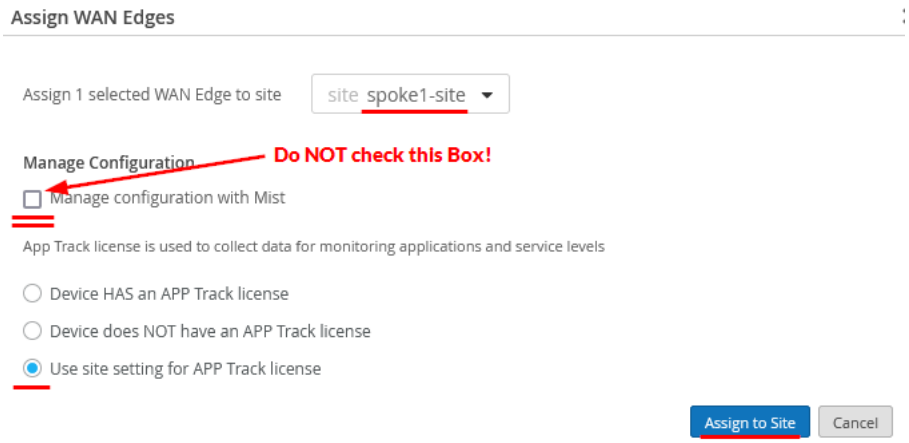
1. In the portal, go to **Organization > Admin > Inventory**.
2. Refresh your browser and check under **WAN Edges** to find out if your SRX Series Firewall is part of the inventory.



3. Assign each SRX Series Firewall to an individual site using the **Assign to Site** option:



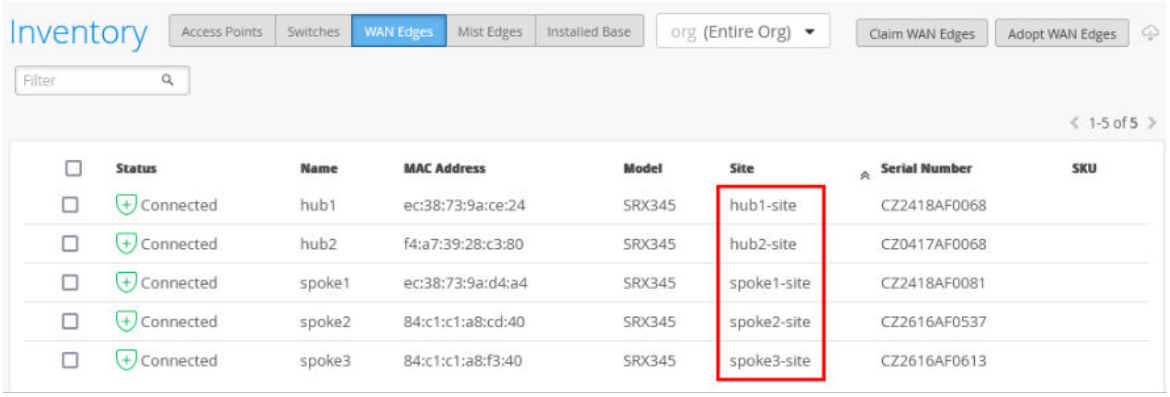
4. On the **Assign WAN Edges** page, select the site you want to assign from the list of available sites.



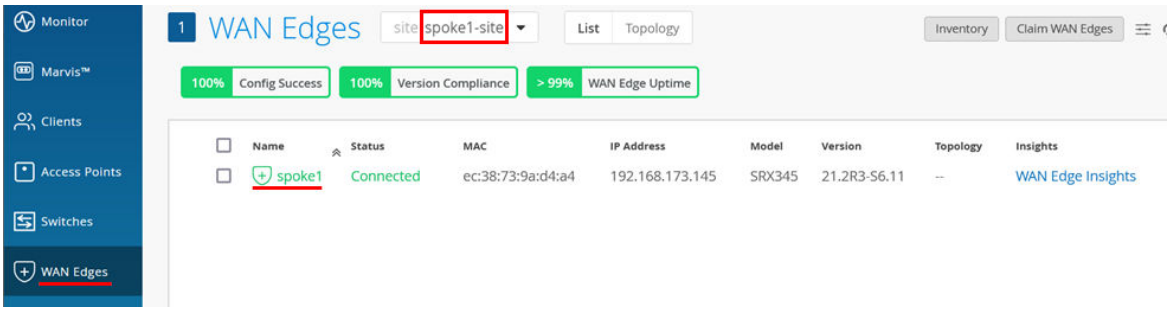
5. Do not select the **Manage configuration with Mist** option. If you do, you may see unwanted changes on your SRX Series Firewall. You can enable the option later if required, after you've assigned the device to the site.

6. Select the **Use site setting for APP Track license** option if you have a valid Application Security license, and then click **Assign to Site**.

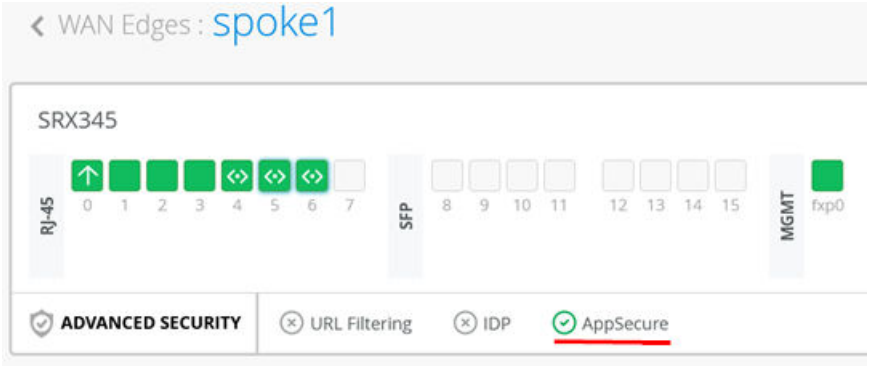
   The figure below shows changes in the inventory once you assign the device to the site:



7. After the device is onboarded, on the **WAN Edges** tab, select **<your site>** and then click on the device:



8. Check the device and AppSecure status.



9. Now, activate **Enable Configuration Management** of the device as the last step so that Juniper Mist can configure the device.

10. (Optional) Use **Remote Shell** to verify the device configuration and status after Juniper Mist cloud takes over management of the device. In our example, you see the output below:

```
root@spoke1> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    up
ge-0/0/0.0              up    up   inet     192.168.173.145/24
.
ge-0/0/1                up    up
ge-0/0/1.0              up    up   aenet    --> ae0.0
ge-0/0/1.1088           up    up   aenet    --> ae0.1088
ge-0/0/1.1099           up    up   aenet    --> ae0.1099
ge-0/0/1.32767          up    up   aenet    --> ae0.32767
ge-0/0/2                up    up
ge-0/0/2.0              up    up   aenet    --> ae0.0
ge-0/0/2.1088           up    up   aenet    --> ae0.1088
ge-0/0/2.1099           up    up   aenet    --> ae0.1099
ge-0/0/2.32767          up    up   aenet    --> ae0.32767
.
ae0                     up    up
ae0.0                   up    up   inet     10.33.33.1/24
ae0.1088                up    up   inet     10.88.88.1/24
ae0.1099                up    up   inet     10.99.99.1/24
ae0.32767               up    up
.
```

```
root@spoke1> show lacp interfaces
Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/0/1 FUP    Actor   No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-0/0/1 FUP  Partner   No   Yes    No   No  Yes   Yes     Fast   Passive
      ge-0/0/2       Actor    No   Yes    No   No   No   Yes     Fast    Active
      ge-0/0/2     Partner    No   Yes    No   No   No   Yes     Fast   Passive
    LACP protocol:        Receive State  Transmit State          Mux State
      ge-0/0/1 FUP              Current   Fast periodic Collecting distributing
      ge-0/0/2                Defaulted   Fast periodic           Detached
```

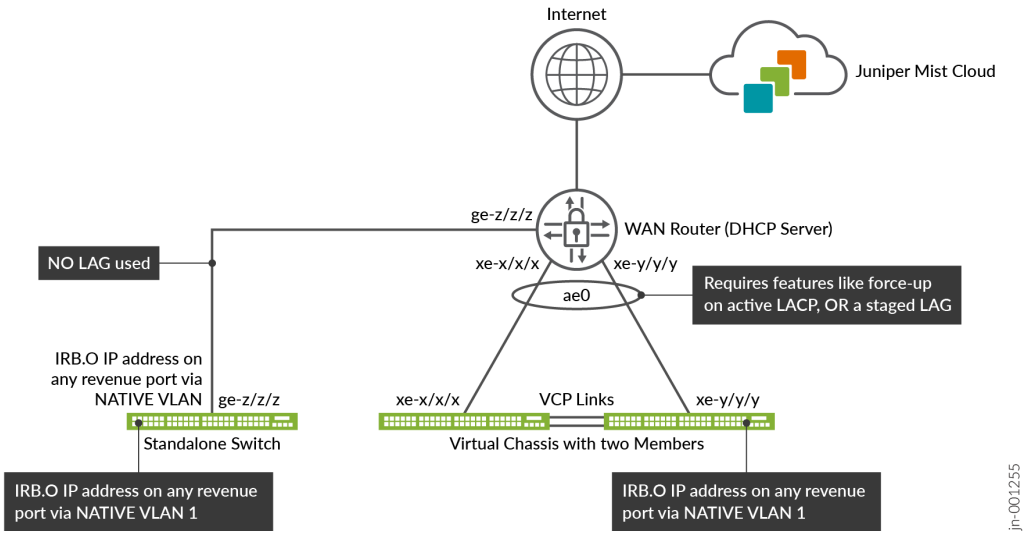
The moment you attach the EX Series Switch and power it up, it should obtain a DHCP lease from the WAN router which you can verify as shown below. From time to time, you should also see the phone-home client on the switch trying to contact the redirect server as in our example:

```
root@spoke1> show dhcp server binding detail
Client IP Address:  10.33.33.11
     Hardware Address:             04:5c:6c:6b:13:42
     State:                        BOUND(LOCAL_SERVER_STATE_BOUND)
     Protocol-Used:                DHCP
     Lease Expires:                2024-03-07 17:02:09 UTC
     Lease Expires in:             85474 seconds
     Lease Start:                  2024-03-06 17:02:09 UTC
     Last Packet Received:         2024-03-06 17:02:09 UTC
     Incoming Client Interface:    ae0.0
     Client Interface Vlan Id:     1
     Server Identifier:            10.33.33.1
     Session Id:                   2
     Client Pool Name:             VLAN1033
root@spoke1> show security flow session source-prefix 10.0.0.0/8
Session ID: 249108247036, Policy name: 01_Internet/20, State: Stand-alone, Timeout: 854, Valid
  In: 10.33.33.11/59874 --> 44.231.144.179/443;tcp, Conn Tag: 0x0, If: ae0.0, Pkts: 8, Bytes:
1278,
  Out: 44.231.144.179/443 --> 192.168.173.145/8983;tcp, Conn Tag: 0x0, If: ge-0/0/0.0, Pkts: 18,
Bytes: 13734,
Total sessions: 1
```

Switch Installation and Configuration

When installing and configuring the Virtual Chassis and standalone switch for the wired part of this lab, we are following the best practice methods shared in more detail in the JVD for Distributed Branch EX Series. In our case, we connect the standalone switch through one trunk port to the WAN router while the Virtual Chassis, for redundancy reasons, utilizes a LAG with force-up configuration on the WAN router (see Figure 23 on page 61). The switch and all attached APs then get a DHCP lease through the native VLAN1033 from the WAN router and are then able to start communicating with the Juniper Mist cloud to be managed. Figure 24 on page 70 shows the intended setup for this lab (without the APs).

Figure 24: In-band Switch and Virtual Chassis MGMT



The following list of steps summarizes the process used to configure the switches in this chapter and is immediately followed by a detailed description of those steps:

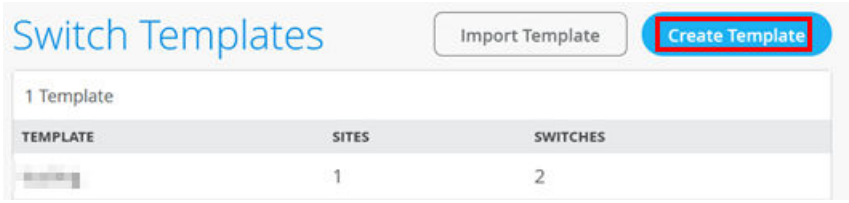
- Define a switch template
- Connect the uplinks to the WAN router
- Claim and ZTP to the Juniper Mist cloud
- Assign the switch to a site
- Optional: Upgrade firmware
- Perform Virtual Chassis formation
- Onboard standalone switches

- Configure access ports for wired clients and APs
- Test the configuration

Define a Switch Template

We start by navigating to **Organization > Switch Templates**:

Then, use **Create Template** or import a JSON file from an existing template:



Using a JSON file, you can import the following configuration to avoid the manual steps described below:

```
{
  "ntp_servers": [],
  "dns_servers": [
    "8.8.8.8",
    "9.9.9.9"
  ],
  "dns_suffix": [],
  "additional_config_cmds": [],
  "networks": {
    "vlan1088": {
      "vlan_id": "1088",
      "subnet": ""
    },
    "vlan1099": {
```

```
      "vlan_id": "1099",
      "subnet": ""
    }
  },
  "port_usages": {
    "dynamic": {
      "mode": "dynamic",
      "rules": []
    },
    "vlan1099-noauth": {
      "mode": "access",
      "disabled": false,
      "port_network": "vlan1099",
      "voip_network": null,
      "stp_edge": true,
      "all_networks": false,
      "networks": null,
      "port_auth": null,
      "speed": "auto",
      "duplex": "auto",
      "mac_limit": 0,
      "persist_mac": false,
      "poe_disabled": false,
      "enable_qos": false,
      "storm_control": {},
      "mtu": null,
      "description": "",
      "disable_autoneg": false,
      "mac_auth_protocol": null,
      "enable_mac_auth": null,
      "mac_auth_only": null,
      "guest_network": null,
      "bypass_auth_when_server_down": null,
      "stp_p2p": false,
      "stp_no_root_port": false,
      "allow_multiple_supplicants": null,
      "dynamic_vlan_networks": null,
      "reauth_interval": null
    },
    "vlan1099-mab": {
      "disabled": false,
      "mode": "access",
      "port_network": "vlan1099",
```

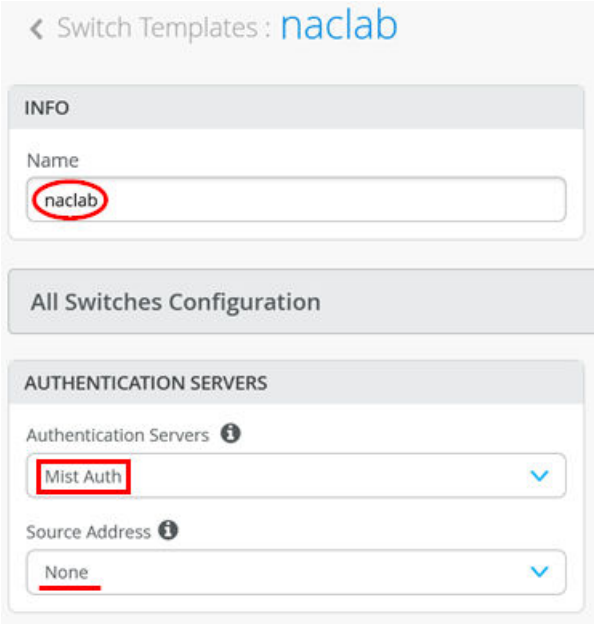
```
      "voip_network": null,
      "stp_edge": true,
      "mac_auth_protocol": "pap",
      "all_networks": false,
      "networks": null,
      "port_auth": "dot1x",
      "allow_multiple_supplicants": true,
      "enable_mac_auth": true,
      "mac_auth_only": true,
      "guest_network": null,
      "bypass_auth_when_server_down": false,
      "dynamic_vlan_networks": null,
      "speed": "auto",
      "duplex": "auto",
      "mac_limit": 0,
      "persist_mac": false,
      "poe_disabled": false,
      "enable_qos": false,
      "storm_control": {},
      "mtu": null,
      "description": "",
      "disable_autoneg": false
    },
    "vlan1099-eap": {
      "disabled": false,
      "mode": "access",
      "port_network": "vlan1099",
      "voip_network": null,
      "stp_edge": true,
      "mac_auth_protocol": null,
      "all_networks": false,
      "networks": null,
      "port_auth": "dot1x",
      "allow_multiple_supplicants": false,
      "enable_mac_auth": false,
      "mac_auth_only": false,
      "guest_network": null,
      "bypass_auth_when_server_down": false,
      "dynamic_vlan_networks": null,
      "speed": "auto",
      "duplex": "auto",
      "mac_limit": 0,
      "persist_mac": false,
```

```
      "poe_disabled": false,
      "enable_qos": false,
      "storm_control": {},
      "mtu": null,
      "description": "",
      "disable_autoneg": false
    }
  },
  "switch_matching": {
    "enable": true,
    "rules": []
  },
  "switch_mgmt": {
    "config_revert_timer": 10,
    "root_password": "juniper123",
    "protect_re": {
      "enabled": false
    },
    "tacacs": {
      "enabled": false
    }
  },
  "radius_config": {
    "auth_servers": [
      {
        "port": "1812",
        "host": "10.44.44.5",
        "secret": "juniper123"
      }
    ],
    "acct_servers": [],
    "auth_servers_timeout": 5,
    "auth_servers_retries": 3,
    "fast_dot1x_timers": false,
    "acct_interim_interval": 0,
    "auth_server_selection": "ordered",
    "coa_enabled": false,
    "coa_port": ""
  },
  "vrf_config": {
    "enabled": false
  },
  "remote_syslog": {
```

```
    "enabled": false
  },
  "snmp_config": {
    "enabled": false
  },
  "dhcp_snooping": {
    "enabled": false
  },
  "acl_policies": [],
  "mist_nac": {
    "enabled": true,
    "network": null
  },
  "port_mirroring": {},
  "disabled_system_defined_port_usages": [],
  "bgp_config": null,
  "routing_policies": {},
  "name": "naclab"
}
```

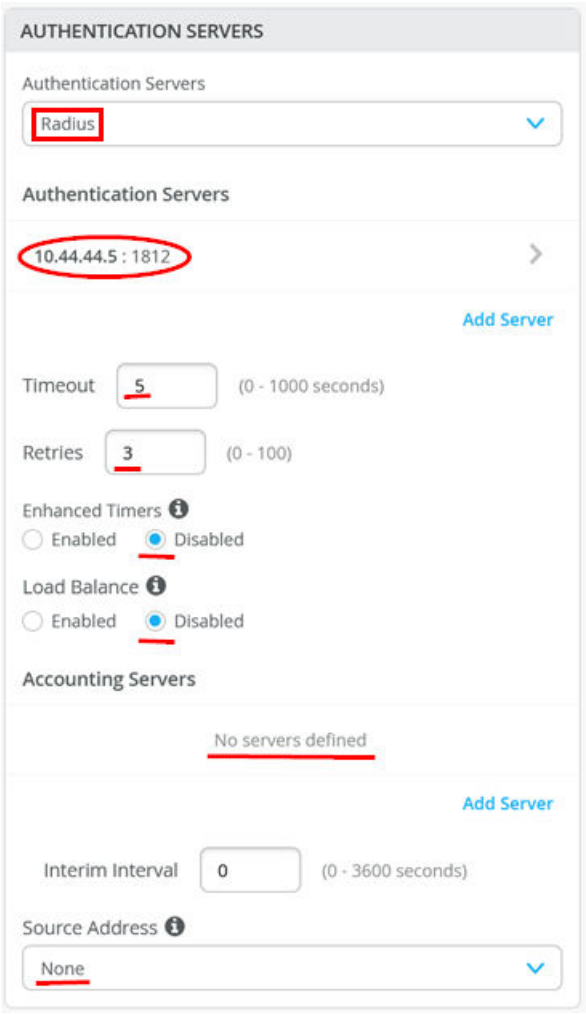

After creating the template, the first and most important step is configuring the switch to use the RadSec tunnel towards the Juniper Mist authentication cloud. Here, you configure the following settings:

- Authentication Servers=`Mist Auth`
- Source Address=`None` (this is the default, and we do not need to change it)



When testing emulated third-party products using a Juniper Mist Edge device as proxy, we make the following changes:

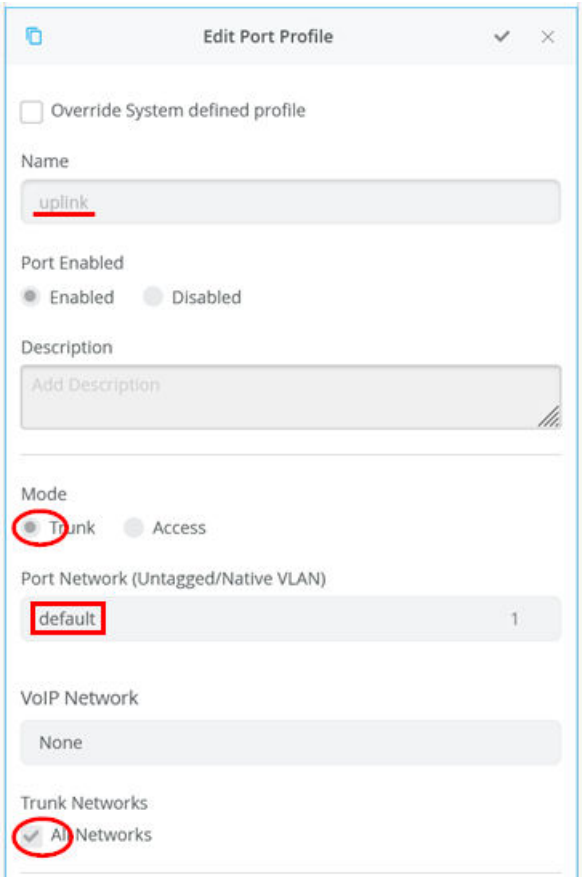
- Authentication Servers=`Mist Auth`
- Auth Server1
  - Hostname / IP Address=`10.44.44.5`
  - Port=`1812`
  - Shared Secret=`juniper123` (or whatever is configured to be used between the two)
- Timeout=`5` (the default)
- Retires=`3` (the default)
- Enhanced Timers=`Disabled` (the default)
- Load Balance=`Disabled` (the default)
- Accounting Servers=`None` (The Juniper Mist Edge does not listen on accounting port 1813)
- Source Address=`None` (this is the default, and we do not need to change it)



Next, define port profiles for uplinks, APs, and authenticating wired clients.

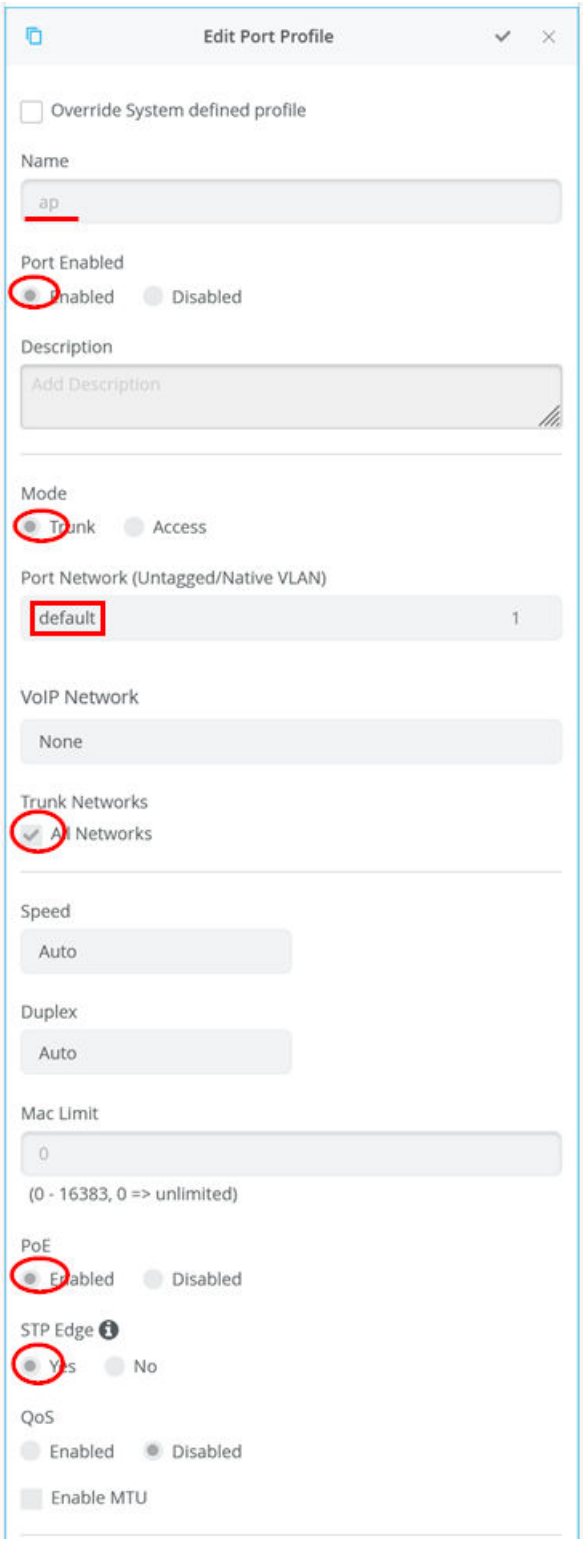
Let's first review two default port profiles we intend to use. The first one is the "uplink" profile where the important settings are:

- Name=`uplink`
- Port Enabled=`Enabled`
- Mode=`Trunk`
- Port Network (Untagged/Native VLAN)=`default` (VLAN-ID=1) (Remember that the WAN router has VLAN1033 configured as the native VLAN. Hence, this enables us to get DHCP leases for the range 10.33.33.0/24 from the WAN router as both the switch and WAN router remove the VLAN tag)
- Trunk Networks=`All Networks` (Allows you to add more VLANs later without needing to change this link)



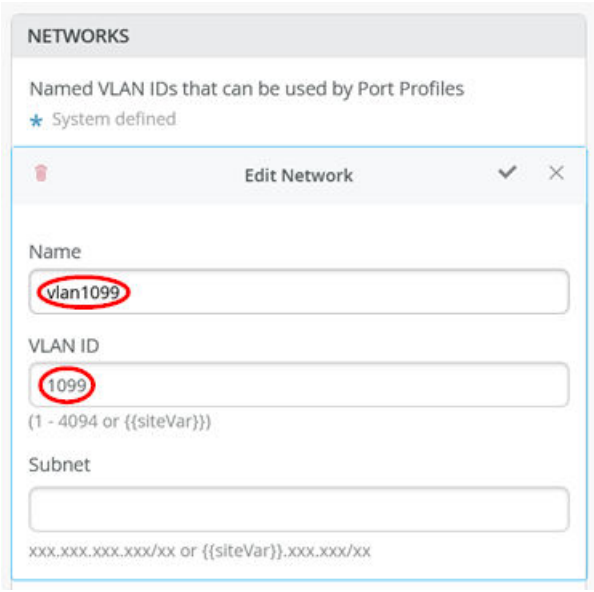
The next default port profile is used for APs:

- Name=`ap`
- Port Enabled=`Enabled`
- Mode=`Trunk`
- Port Network (Untagged/Native VLAN)=`default` (VLAN-ID=1) (Here, we are stitching the native uplink VLAN from WAN router further to the AP to manage the AP)
- Trunk Networks=`All Networks`
- PoE=`Enabled`
- STP Edge=`Yes`



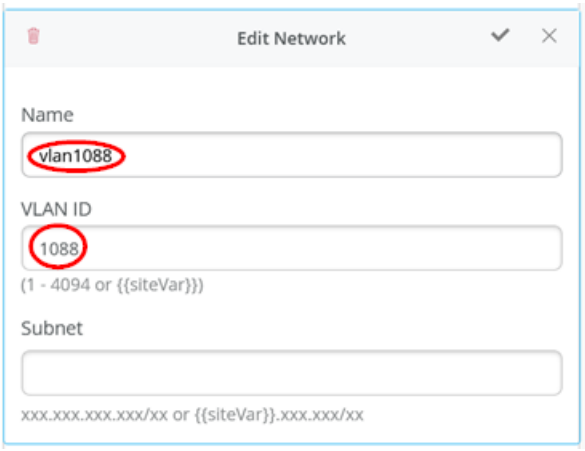
Before we can continue, we need to define the three VLANs we use in the switching and AP environment. VLAN1033 does not need to be defined as we continue using the existing default VLAN1. The first network is the one for wired clients. So, we configure:

- Name=`vlan1099`
- VLAN-ID=`1099`
- Subnet=`Empty` (Enter “10.99.99.0/24” here when using a campus fabric).



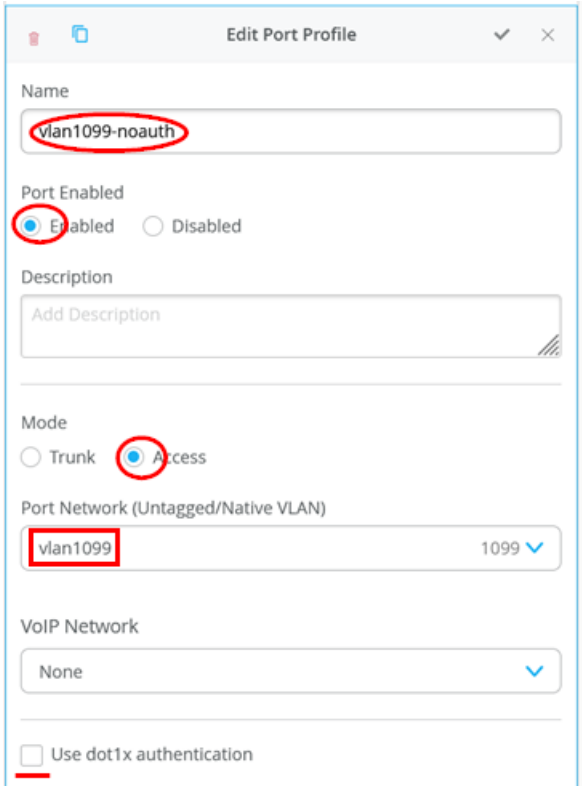
The second network is the transport for wireless clients. So, we configure:

- Name=`vlan1088`
- VLAN-ID=`1088`
- Subnet=`Empty` (Enter “10.88.88.0/24” here when using a campus fabric).



Next, we configure for wireless clients using VLAN1099 three different port profiles for testing. We start with one that has no authentication at all so we can use it for client connectivity testing without Access Assurance.

- Name=`vlan1099-noauth`
- Port Enabled=`Enabled`
- Mode=`Access`
- Port Network (Untagged/Native VLAN)=`vlan1099`
- Use dot1x authentication=`Unchecked`

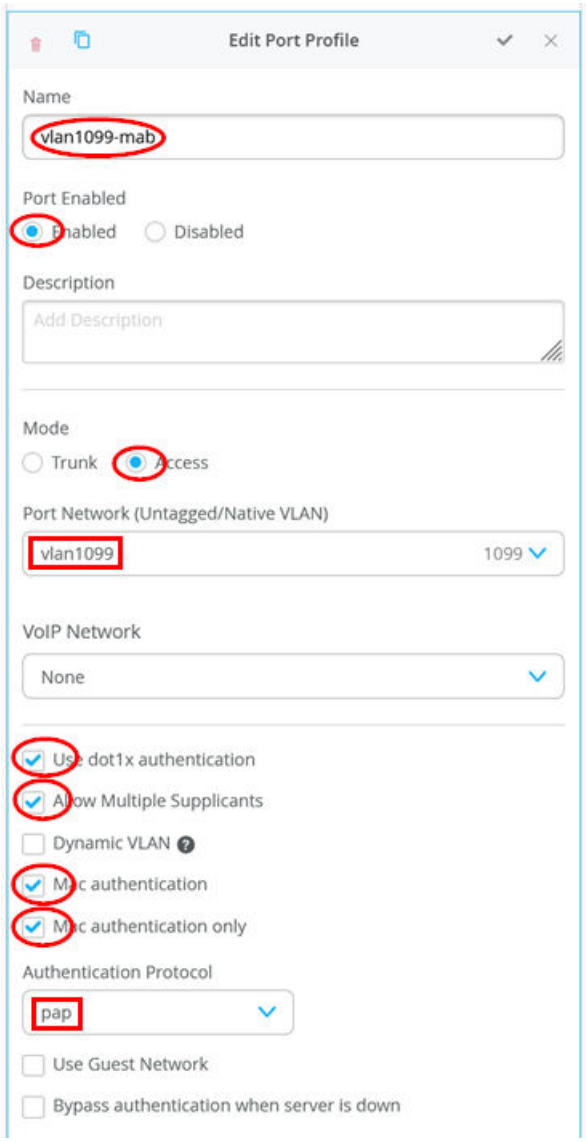


The next port profile is used if the attached client does not support EAP, so we have to fall back to MAC address-based authentication only.

- Name=`vlan1099-mab`
- Port Enabled=`Enabled`
- Mode=`Access`
- Port Network (Untagged/Native VLAN)=`vlan1099`
- Use dot1x authentication=`Checked`
- Allow Multiple Supplicants=`Checked` (This recommended setting is better enabled for labs as sometimes multiple MAC addresses appear on a port, and you do not want unknown MAC addresses to block further authentication on the port).

- Dynamic VLAN=Unchecked
- Mac Authentication=Checked (Enables MAB)
- Mac Authentication only=Checked (Disables all EAP authentication for this port, otherwise you must wait 60 seconds)
- Authentication Protocol=pap (This is a best practice setting. This has no effect if the switch uses a RadSec tunnel. This setting defines the used MAB RADIUS authentication for third-party RADIUS servers or Mist-Edge as RadSec-Proxy)
- Use Guest Network=Unchecked
- Bypass authentication when server is down=Unchecked

Figure 25: Wired MAB Profile

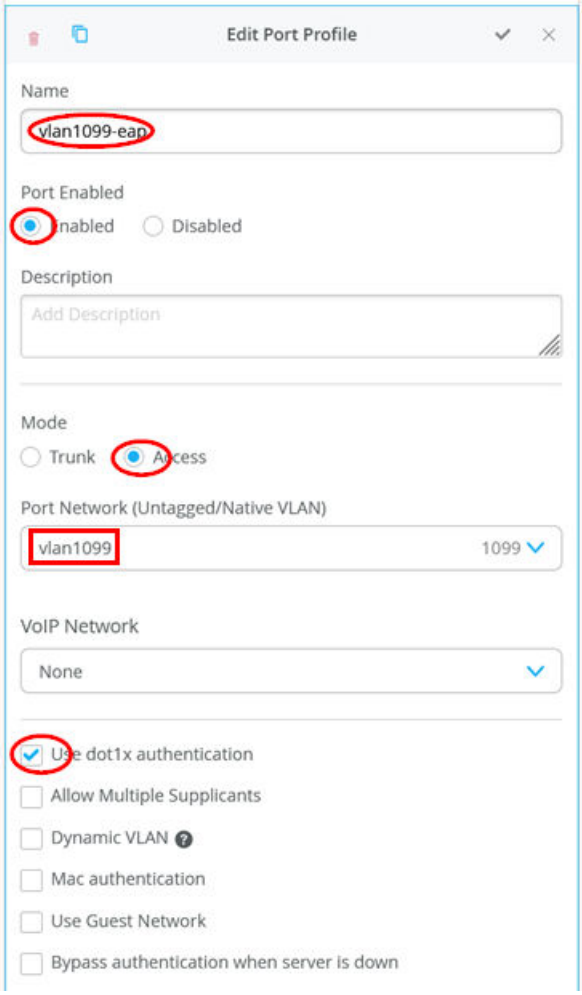


Finally, the last port profile is used for EAP supplicants, which should be the best practice for all clients we have in the environment:

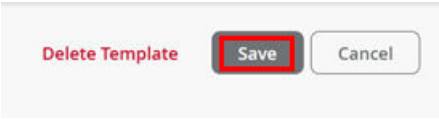
- Name=`vlan1099-eap`
- Port Enabled=`Enabled`
- Mode=`Access`
- Port Network (Untagged/Native VLAN)=`vlan1099`
- Use dot1x authentication=`Checked`

- Allow Multiple Supplicants=Unchecked
- Dynamic VLAN=Unchecked
- Mac Authentication=Unchecked
- Mac Authentication only=Unchecked
- Use Guest Network=Unchecked
- Bypass authentication when server is down=Unchecked

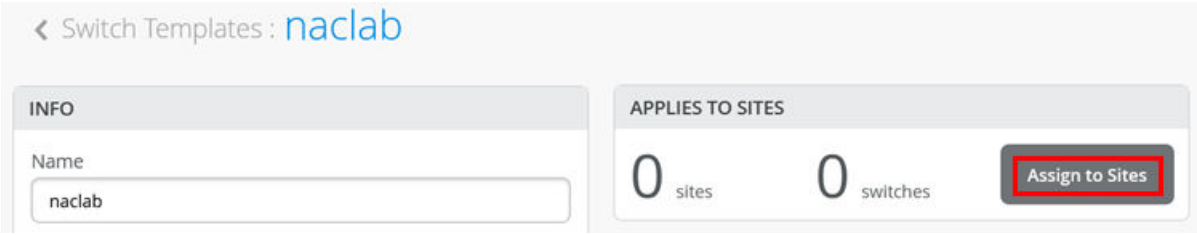
Figure 26: Wired EAP Profile



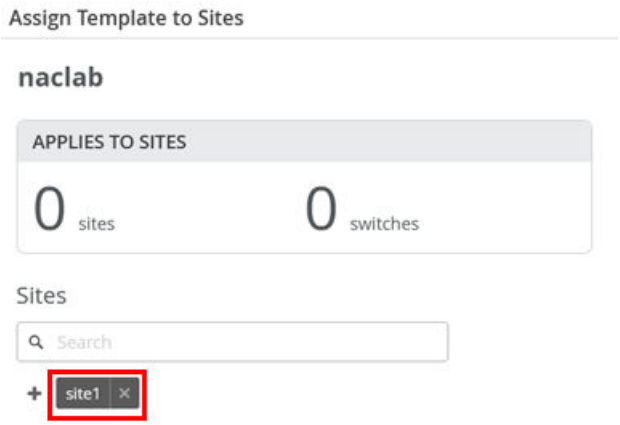
Do not forget to save your template.



After saving the template, assign it to a site where you intend to use it.



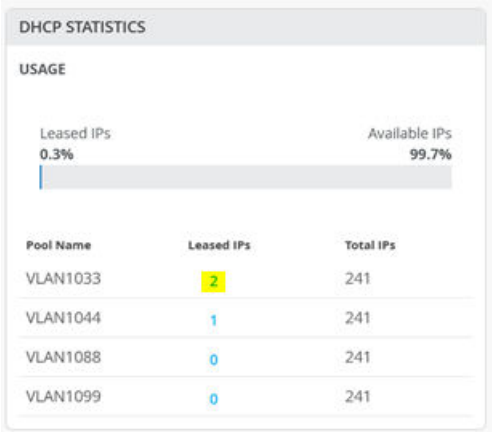
Here, we add “site1” to our template before we apply it.



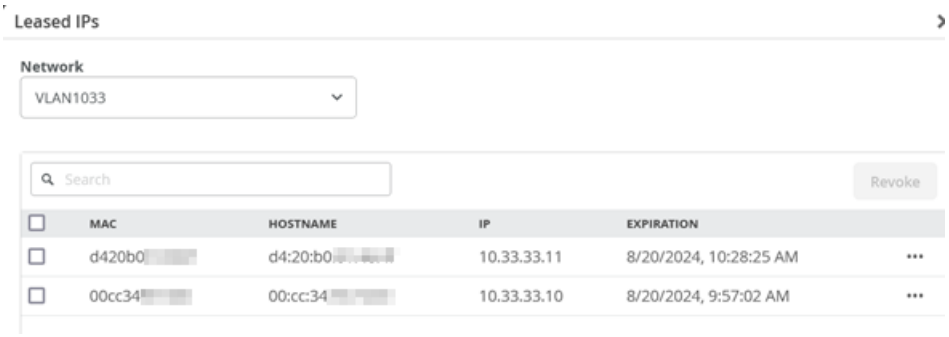
Resulting in the below example:

Connect the Uplinks to the WAN Router

In this step, use the revenue ports on the switches and connect them to the WAN router. Power on the switches afterward. After about 5 minutes, they are booted up and should request DHCP leases.



You can review the DHCP lease list to see more information:

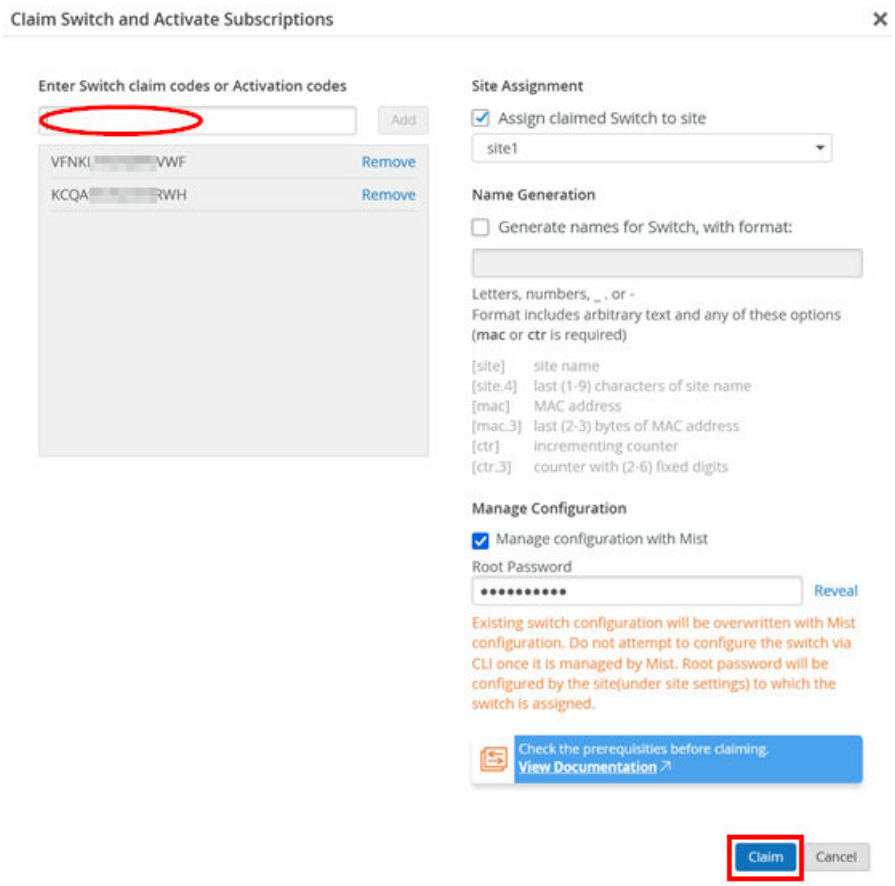


Claim and ZTP to Juniper Mist Cloud

There are multiple methods of onboarding a switch to the Juniper Mist cloud. They are all described in the Distributed Branch EX Series JVD. To simplify things for this lab, we use the claim and ZTP method. The process of ZTP is described here for review. The process in our case is described by the following:

- Locate the QR label on the device and scan or read the claim code.
- Go to **Organization > Inventory**.
- Select **Switches** and then click on **Claim Switches**, similar to what you see in Figure 27 on page 87:

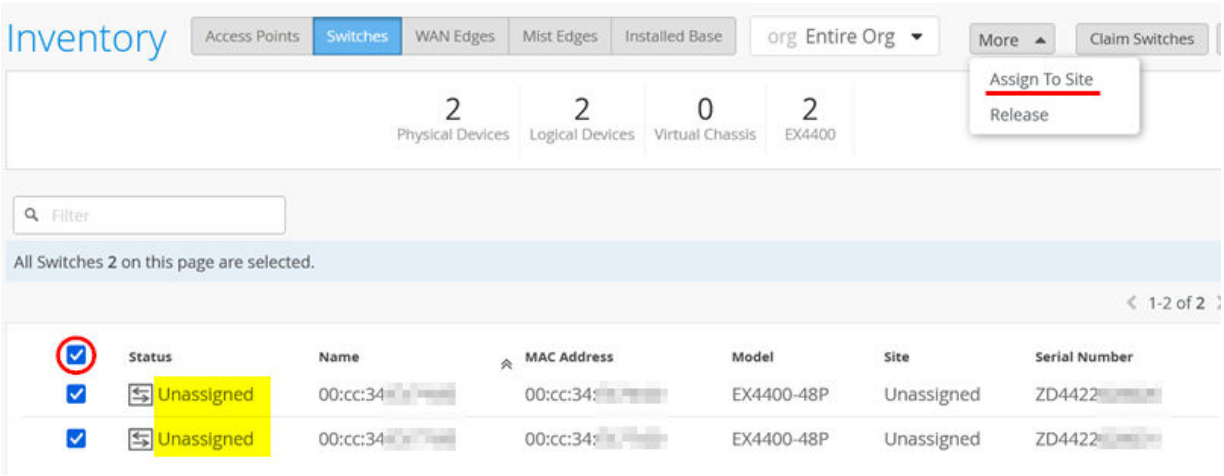
Figure 27: Claim and Assign to Site



Assign to Site

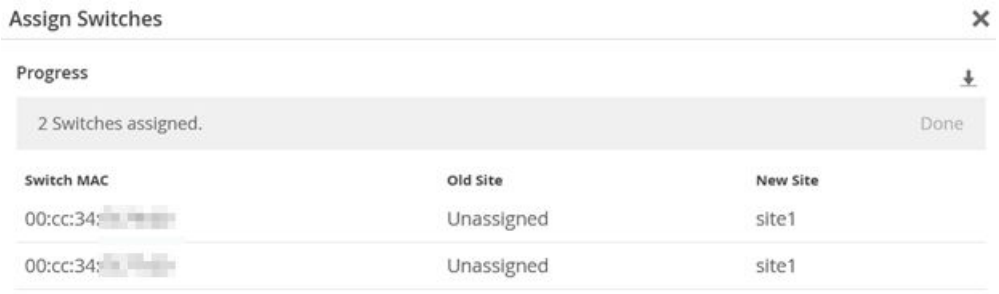
If you have not already assigned the switches to a site as shown in Figure 27 on page 87, remember that you must first assign switches to a site before you can manage them. As seen in Figure 28 on page 88, select the switches and then from the **More** menu, select **Assign To Site**.

Figure 28: Assign Switches to a Site

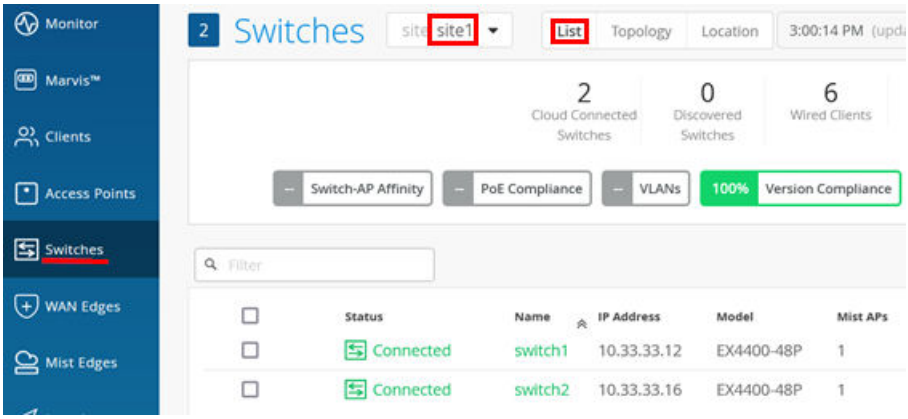


Select the site where these switches will be used and enable **Manage configuration with Mist** before you click on **Assign to Site**.

Something like the figure below should be displayed:



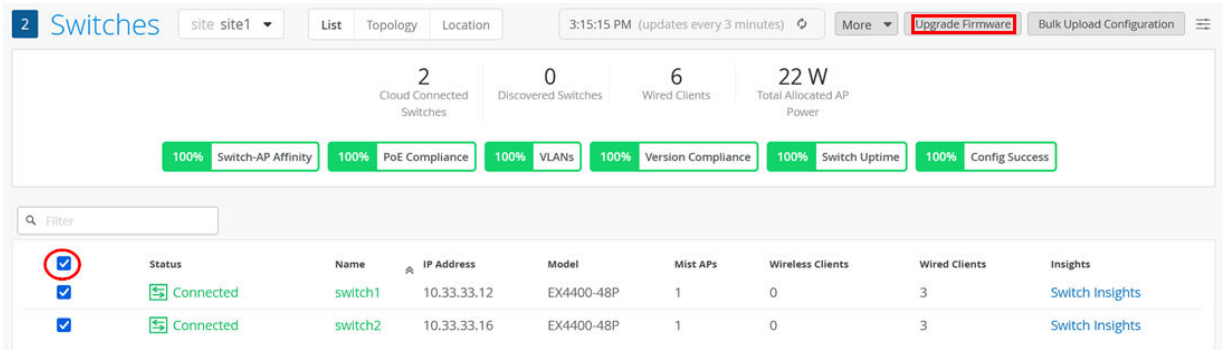
After the switches receive their DHCP lease from the WAN router and get redirected to the correct Juniper Mist cloud, we should see them appearing in the “Connected” state after navigating to **Switches > select <site> > List**.



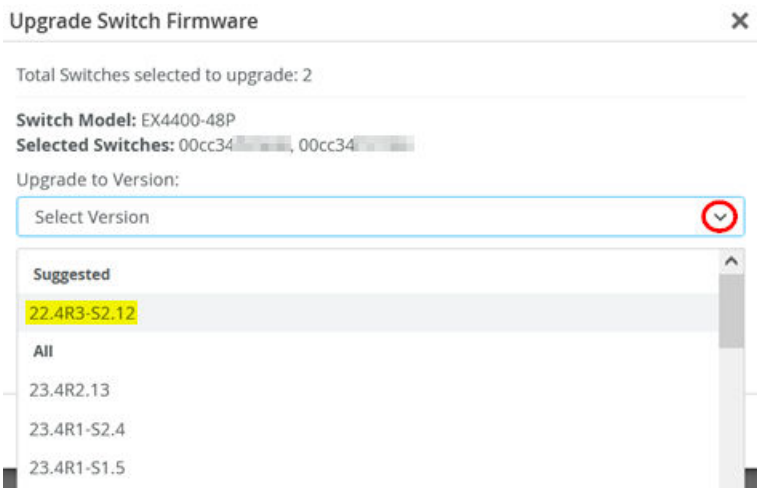
Optional: Upgrade Firmware

> **NOTE**: When planning to form a Virtual Chassis, it is recommended that all switch members have the same Junos OS firmware version running. Follow the upgrade procedure if that is not the case.

It is suggested that you upgrade the Junos firmware to the suggested version before you put the switch into production. To achieve this, select the switches then select **Upgrade Firmware** as shown in the figure below:



This will open a dialogue where you can select the Junos OS firmware under **Suggested** for your switch model as shown in the figure below:



Further information about this process is given in the JVD for Distributed Branch EX Series in this chapter.

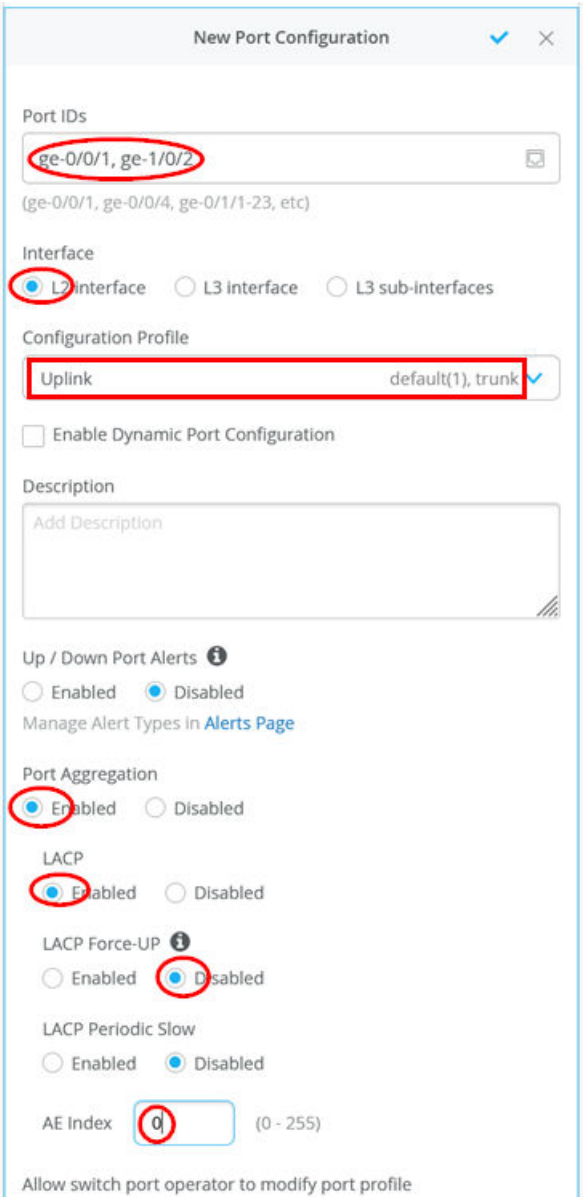
Perform a Virtual Chassis Formation

If you plan to build a Virtual Chassis, you can find the instructions for how to do so for each switch model in the JVD for Distributed Branch EX Series. The following links are for the most common switch models:

- Switches that support ZTP: Workflow for VC formation with Mist for EX3400, EX4100, EX4100-F, EX4300, EX4400 & EX4600
- Switches that need to be pre-staged for Virtual Chassis formation: Workflow for Virtual Chassis Formation with Mist for EX2300, EX4650 and QFX5120

After the Virtual Chassis has been formed, configure the LAG uplink. In our case, configure the following:

- Port IDs=`ge-0/0/1, ge-1/0/2`
- Interface=`L2 Interface`
- Configuration Profile=`Uplink`
- Port Aggregation=`Enabled`
- LACP=`Enabled`
- LACP Force-UP=`Disabled` (This option is only needed for downlink interfaces such as those on the WAN router)
- AE Index=`0`



(Optional) You can **Remote Shell** to the switch to see the LAG and LACP states like in this example output:

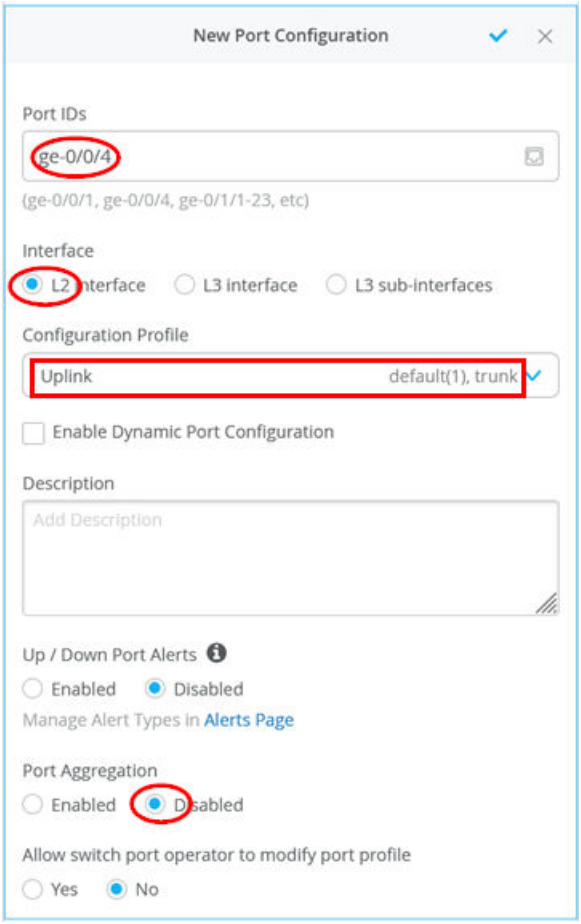
```
root@switch1> show lacp interfaces
Aggregated interface: ae0
    LACP state:          Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/0/1           Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-0/0/1         Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-1/0/2           Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-1/0/2         Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
```

```
      ge-0/0/1                Current   Fast periodic Collecting distributing
      ge-1/0/2                Current   Fast periodic Collecting distributing
```

Onboard Standalone Switches

Onboarding the standalone switches involves using the previously demonstrated steps for bringing the switch online and managing it in Juniper Mist cloud. Once this is done, merely configure the uplink interface towards the WAN router with the **Uplink** profile since multiple VLANs must be supported on this link. In our example, the configuration is:

- Port IDs=`ge-0/0/4`
- Interface=`L2 Interface`
- Configuration Profile=`Uplink`
- Port Aggregation=`Disabled`



(Optional) You may want to use **Remote Shell** to determine the VLANs configured on port ge-0/0/4.

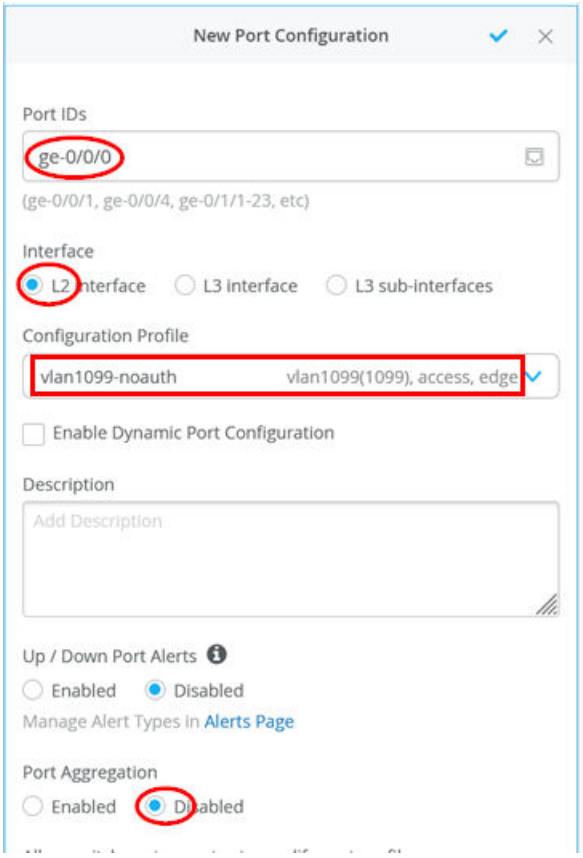
```
root@switch2> show vlans
Routing instance        VLAN name             Tag          Interfaces
default-switch          default               1
.
.
                                                           ge-0/0/4.0*
.
.
default-switch          vlan1088              1088
                                                           ge-0/0/4.0*
default-switch          vlan1099              1099
                                                           ge-0/0/4.0*
```

Configure Access Ports for Wired Clients and APs

Next, configure the access ports for the wired clients and APs. We suggest doing that by using switch templates to synchronize configurations better. For our lab however, individually assigned profiles make more sense since we are more flexible with changing them.

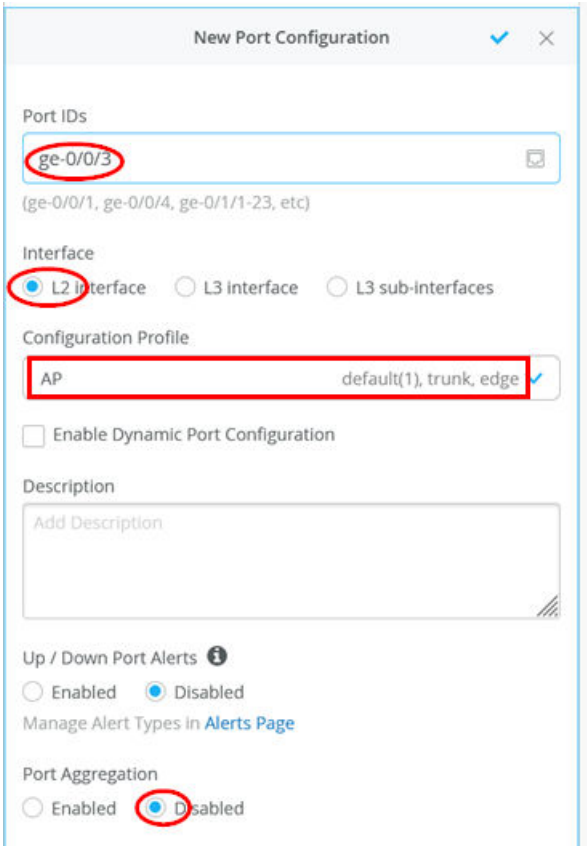
For the wired client using VLAN 1099, we start with the “vlan1099-noauth” profile like that shown below:

- Port IDs=`ge-0/0/0`
- Interface=`L2 Interface`
- Configuration Profile=`vlan1099-noauth`
- Port Aggregation=`Disabled`



For the AP, we leverage the built-in AP profile:

- Port IDs=`ge-0/0/3`
- Interface=`L2 Interface`
- Configuration Profile=`AP`
- Port Aggregation=`Disabled`



Test Configuration

Next, check whether the wired clients are connected to the infrastructure and can see each other without any authentication performed yet. Choose whatever method is available in your lab to connect to a wired client attached to your switches. Below, see commands executed on a Linux client testing the connectivity in the network, for example:

```
# review the local interfaces
root@desktop1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
1: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
1000
    link/ether 52:54:00:7a:8a:50 brd ff:ff:ff:ff:ff:ff
    inet 10.99.99.99/24 brd 10.99.99.255 scope global ens5
       valid_lft forever preferred_lft forever
```

```
    inet6 fe80::5054:ff:fe7a:8a50/64 scope link
       valid_lft forever preferred_lft forever
# review the routes
root@desktop1:~# ip r
default via 10.99.99.1 dev ens5 proto static
10.99.99.0/24 dev ens5 proto kernel scope link src 10.99.99.99
# test the connection via WAN-Router to internet
root@desktop1:~# ping -c3 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=53 time=3.65 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=53 time=3.54 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=53 time=3.63 ms
.
# ping the wireless client on the same VLAN in the next Switch
root@desktop1:~# ping -c3 10.99.99.42
PING 10.99.99.42 (10.99.99.42) 56(84) bytes of data.
64 bytes from 10.99.99.42: icmp_seq=1 ttl=64 time=1.47 ms
64 bytes from 10.99.99.42: icmp_seq=2 ttl=64 time=0.676 ms
64 bytes from 10.99.99.42: icmp_seq=3 ttl=64 time=0.679 ms
.
# check the ARP-resolution
root@desktop1:~# ip n
10.99.99.42 dev ens5 lladdr 52:54:00:bd:8c:e8 STALE
10.99.99.1 dev ens5 lladdr 4c:96:14:55:7f:80 STALE
# test the DHCP-Server on WAN-Router
root@desktop1:~# dhclient -v ens5
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/
Listening on LPF/ens5/52:54:00:7a:8a:50
Sending on   LPF/ens5/52:54:00:7a:8a:50
Sending on   Socket/fallback
DHCPDISCOVER on ens5 to 255.255.255.255 port 67 interval 3 (xid=0xf4b2324f)
DHCPOFFER of 10.99.99.10 from 10.99.99.1
DHCPREQUEST for 10.99.99.10 on ens5 to 255.255.255.255 port 67 (xid=0x4f32b2f4)
DHCPACK of 10.99.99.10 from 10.99.99.1 (xid=0xf4b2324f)
bound to 10.99.99.10 -- renewal in 35575 seconds.
# test the DNS resolution
root@desktop1:~# host www.google.com
www.google.com has address 172.217.164.100
www.google.com has IPv6 address 2607:f8b0:4005:80b::2004
```

Access Point Installation and Configuration

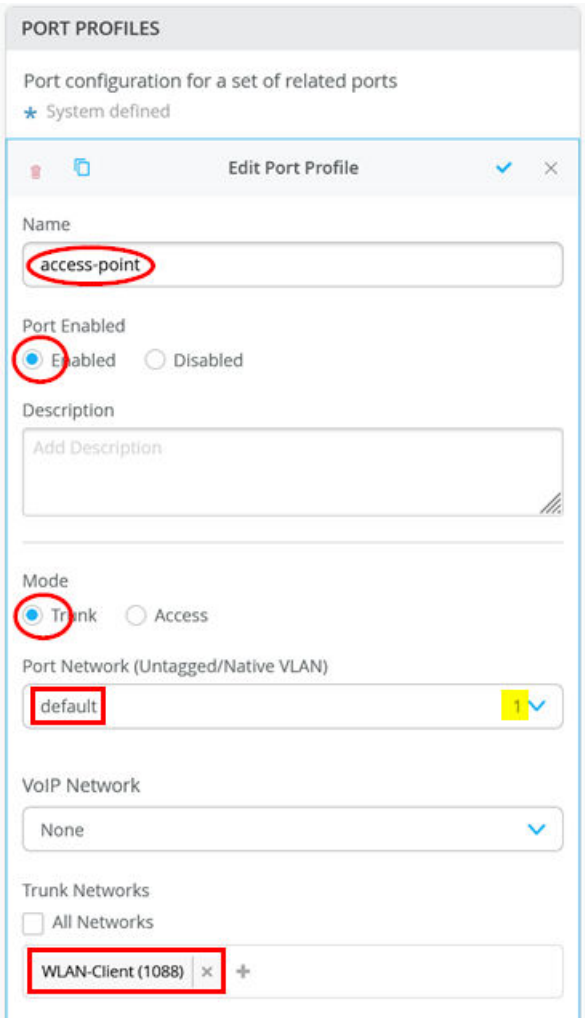
Pre-conditions That Must be Met for AP Onboarding

To be able to execute this lab, we assume that the following conditions are met:

- The AP is cabled to the EX Series Switch and has power (through an external power supply or through PoE).
- If PoE is used, the switch must have enough PoE power available and must support, at a minimum, IEEE 802.3af or IEEE 802.3at depending on your AP model.
- Link Layer Discovery Protocol (LLDP) should be activated (this is not mandatory) on switches and routers for debugging.
- It is necessary to have completely followed the instructions from the above chapters as those include instructions for the following:
  - A DHCP server is configured for the AP management VLAN 1033 subnet 10.33.33.0/24 in this branch. In our example, that is done on the WAN router.
  - An additional VLAN 1088 subnet 10.88.88.0/24 is used for WLAN clients.
  - An additional VLAN 1099 subnet 10.99.99.0/24 is used for wired clients.
  - On the WAN router, a LAG is formed to the switch containing native VLAN 1033 and VLANs 1088 and 1099 as trunk. Keep in mind that without further changes, the native VLAN from the uplink gets assigned to VLAN 1 on the switch where irb.0 for in-band management is assigned. Providing that management VLAN to a downstream device such as an AP needs to reference this VLAN 1.
  - On the port where the switch is attached, the default VLAN 1 is native and VLAN 1088 is trunked. We did not add VLAN 1099 for wired clients again as the port does not need to support it.
  - The WAN router must implement some form of source NAT on the WAN interfaces to allow management traffic towards the Juniper Mist cloud.

Below is just a reminder about the minimal configuration on the switch (apart from the uplink):

Figure 29: Access-Point Port Profile on Switch

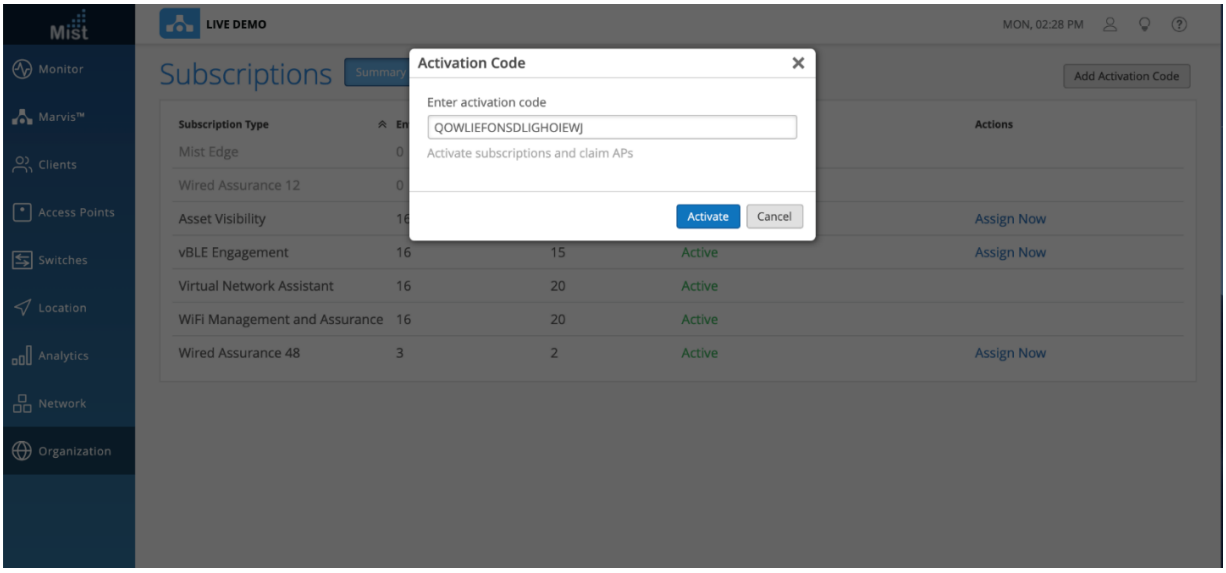


How to Claim APs to an Organization?

APs can be claimed to any organization by using either an activation code, claim code, or QR code.

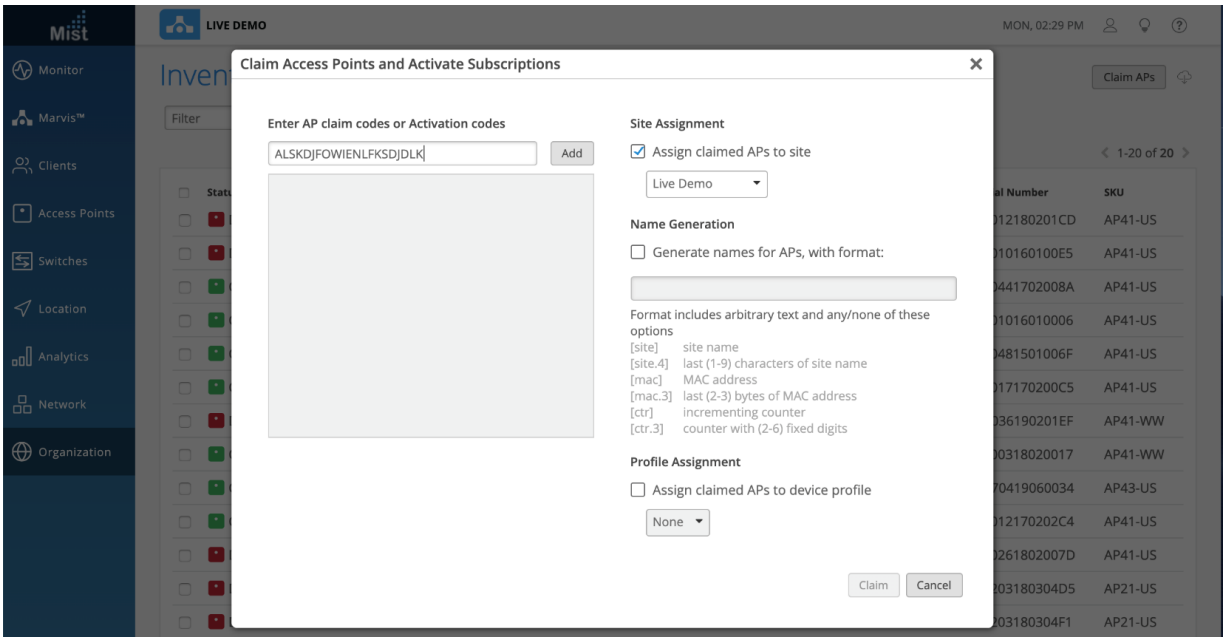
Example: Activation Code

Whenever you order APs, our sales operations team will send you an activation code which can be used for claiming the APs and subscriptions as per the purchase order. You can use this activation code to claim APs in one go. To claim the APs, go to the **Organization > Subscriptions** page and select **Add Activation Code** in the upper-right corner. Once the activation code is added and activated, all the APs will automatically get claimed to the organization. You can see the list of APs on the Inventory page (**Organization > Inventory**).



Example: AP Claim Code

You can claim individual APs to your organization by navigating to **Organization > Inventory > Claim APs** and entering in the claim code found on the back of each AP.



Example: AP QR Code

Using the Mist™ AI mobile app, you can scan the QR code printed on the back of Juniper APs to claim APs to your organization. Our app is compatible with both iOS and Android devices. Read more about it here.

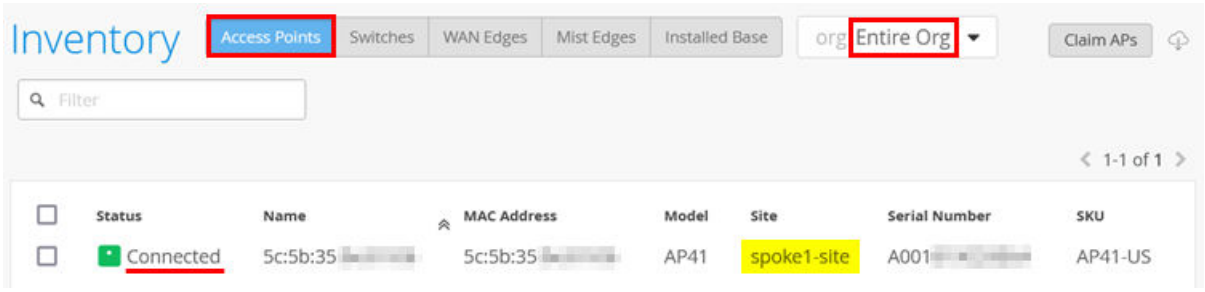
Where Can I Get the Claim Codes for the APs in My Organization?

The claim code of the AP is written on the back of the AP where the QR code of the AP is printed.

Troubleshoot: Bringing the AP Online as “Connected” into the Inventory

We assume in this step you have used any of the methods above to claim the AP into the inventory. When you refresh the browser window after 3-5 minutes, the AP should appear automatically in the “Connected” state in the inventory such as seen in Figure 30 on page 100:

Figure 30: Access Point status



If that is the case, you can skip this section since it describes how to troubleshoot if the AP does not appear as “Connected”.

The troubleshooting of Juniper APs is explained in detail here. This remaining section will focus on what to troubleshoot on the EX Series Switch side to help bring the AP into the “Connected” state in the inventory.

NOTE: If the AP is not assigned to a site, it will always appear in the “Disconnected” state irrespective of its connectivity to the cloud. Remember that along with claiming the AP, assigning the AP to a site is mandatory.

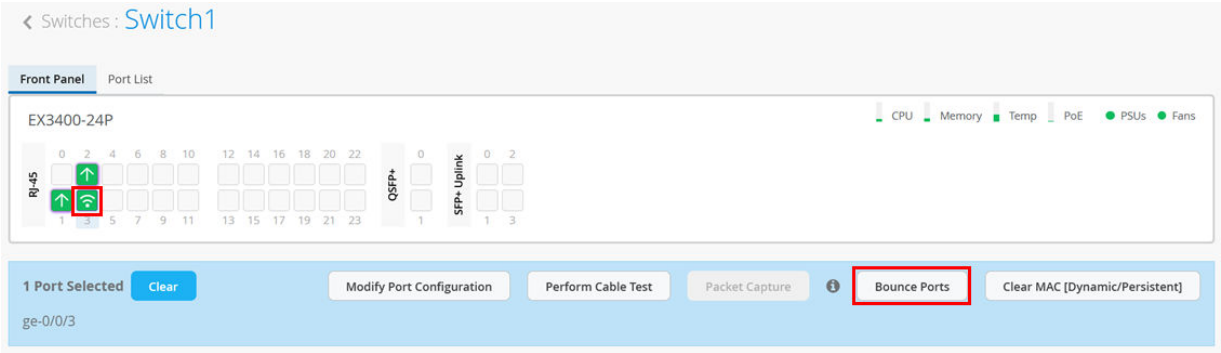
If you are local to the site, check the AP’s status LED since its blinking code may indicate the error by cross referencing it with Figure 31 on page 101.

Figure 31: LED Blink Patterns

LED Blink Pattern	AP Status
	AP starting to boot - 3 seconds
	AP booting - 12 seconds
	Connecting to cloud - 30-40 seconds
	AP connected to the cloud
	AP configured by the cloud
	At least one wireless client connected
	AP upgrading
	'Locate the AP' option in the GUI
	AP failure mode
	User holding down the 'Reset to factory default' button
	AP is going to reset
	Insufficient PoE power
2	No ethernet link - Has no link (Seen using power injectors, but not connected to a switch)
3	No IP Address - No DHCP lease or no static IP in config (DHCP server not configured and/or working)
4	No default gateway - No default gateway in DHCP lease or in static config
5	Default gateway unreachable - No ARP response from default gateway
6	No DNS - No DNS server(s) in DHCP lease or in static config
7	No DNS response - No response to DNS lookup (received DNS server via DHCP but can not reach or ping Mist Cloud)
8	Empty DNS response - DNS response contains no address records
9	Duplicate IP Address - Duplicate IP address detected on the LAN (ARP probes)
1,2	Cloud unreachable - TCP SYN fails & cannot ping ep-terminator
1,3	No cloud response - Ping-able, but TCP port 443 doesn't go through
1,4	Cloud cert time check failed - NTP time is not within cert's not-before/not-after times
1,5	Cloud cert invalid - Some 'security' interceptor is messing with the cert (aka the stolen CA cert + snoopy IT)
1,6	Mutual auth failed - Mutual authentication is failing between the AP and cloud
1,7	Config fetch failed - The cloud is unable to provide the AP with a configuration
1,8	Invalid configuration - The cloud provided an invalid configuration
1,9	Boot config save failed - Unable to save (or delete) the boot config
2,1	L2TP management tunnel peer unreachable - SCCRQ fails & L2TP management server is not pingable
2,3	No response from L2TP management tunnel peer - L2TP management server is pingable, but sends no response to our SCCRQ
2,4	L2TP management tunnel config rejected - L2TP management credentials failed. Our SCCRQ returns a StopCCN instead of SCCRP
2,5	L2TP management tunnel stopped - L2TP management server sent us a StopCCN and killed the tunnel
2,6	L2TP management session config rejected - L2TP management server sends CDN in response to our ICRQ
2,7	L2TP mgmt session shutdown - L2TP management server sent us a CDN and killed the session
3,1	L2TP DHCP no response - No response to our DHCP discover over the management tunnel
3,2	L2TP default gateway missing - DHCP offer lacks a default gateway
3,4	L2TP default gateway unreachable - No ARP response from default gateway
3,5	L2TP management DNS missing - DHCP offer lacks any DNS servers
4,1	Boot config unreadable - Configuration file exists, but it is not readable
4,2	Boot config invalid - Contents of the boot configuration weren't usable
4,3	Boot config failed - Contents caused loss of connection to the cloud
5,1	Firmware corrupt - Damaged firmware image (missing/unreadable files). Try another image, and/or re-image
5,2	Unexpected failure - Some API which should not have failed, did
6,1	Proxy config invalid - The proxy configuration is not usable
6,2	Empty DNS response to proxy host lookup - DNS response contains no A (address) records for the proxy host
6,3	Proxy is unreachable - Can't contact proxy
6,4	No proxy server response - Can ping proxy, but can't connect to proxy TCP port
6,5	Proxy Authentication Required - Proxy returns code 407; Authentication required

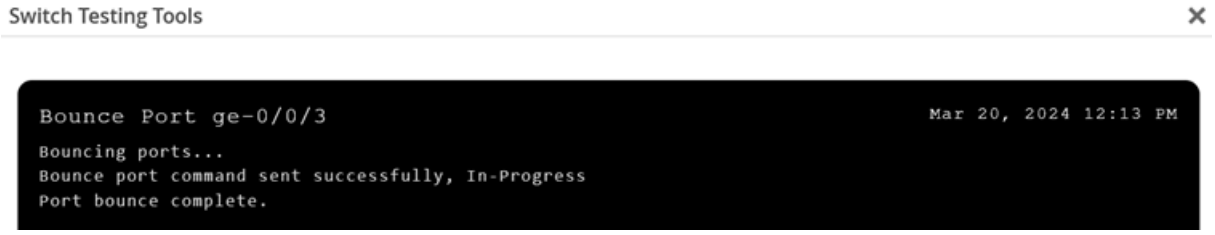
The first thing to do is to understand the source of the error by viewing the AP's status LED. If you can't see the LED on the AP, reboot it. If the AP is powered by PoE, then you can reboot it by using one of the two following methods:

- Use the switch's **Bounce Ports** option found on the portal. Selecting the port where the AP is attached opens a pane where you can choose to bounce a particular port which will also power cycle the attached AP.



You will see a new window with information about the bounce port status as shown in Figure 32 on page 102:

Figure 32: Bounce a Port



NOTE: Bouncing a port may take several minutes since two Junos OS configuration commits must happen as part of the process.

- Change the PoE interface configuration using **Remote Shell** (This option is not recommended).

```
cli
edit
set poe interface ge-0/0/3 disable
commit
# wait >20seconds
delete poe interface ge-0/0/3 disable
commit and-quit
exit
```

If the AP is powered by an external power supply, you must unplug it for a while and then power it on again. The AP has no console connection like the switches.

The next step is to use **Remote Shell** to the switch to review items such as:

- Is the port that the AP is connected to showing as “up”?
- Is the port that the AP is connected to correctly configured and forwarding packets?
- Does the MAC address of the AP appear on the expected port?
- Do you see the AP appearing as an LLDP neighbor?
- Assuming the AP is powered by PoE, what’s the actual power draw?

The following example output shows an AP that is attached to interface ge-0/0/3:

- Check if the port is administratively “up” and a physical link detected.

```
root@Switch1> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    down
ge-0/0/0.0              up    down eth-switch
ge-0/0/1                up    up
ge-0/0/1.0              up    up   aenet    --> ae0.0
ge-0/0/2                up    up
ge-0/0/2.0              up    up   aenet    --> ae0.0
ge-0/0/3                up    up
ge-0/0/3.0              up    up   eth-switch
ge-0/0/4                up    down
ge-0/0/4.0              up    down eth-switch
.
```

- Then, check that the port is in forwarding mode and that it is configured with the expected VLAN (the default VLAN in this example) and is in access mode. Also, verify that the VLAN assigned is the same VLAN that the WAN router uses for DHCP lease handouts for AP management.

```
root@Switch1> show ethernet-switching interface ge-0/0/3
Routing Instance Name : default-switch
Logical Interface flags (DL - disable learning, AD - packet action drop,
                         LH - MAC limit hit, DN - interface down,
                         MMAS - Mac-move action shutdown,  AS - Autostate-exclude enabled,
                         SCTL - shutdown by Storm-control, MI - MAC+IP limit hit)
Logical         Vlan                  TAG   MAC    MAC+IP STP         Logical
Tagging
interface       members                     limit  limit  state       interface flags
ge-0/0/3.0                                  32768  0                                tagged
                WLAN-Client           1088  32768  0      Forwarding                tagged
```

```
                default                 1      32768  0      Forwarding
untagged
```

- Next, check if the MAC address (also printed on the QR code) of the AP appears on the switch's interface as expected.

```
root@Switch1> show ethernet-switching table
MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static, C -
Control MAC
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)
Ethernet switching table : 2 entries, 2 learned
Routing instance : default-switch
    Vlan                MAC                 MAC         Age    Logical
NH        RTR
    name                address             flags              interface
Index     ID
    default             5c:5b:35:be:81:06   D             -    ge-0/0/3.0
0         0
    default             ee:38:73:9a:d4:a5   D             -    ae0.0
0         0
```

NOTE: At this point, it is appropriate to check the port authorization status of the AP as well. Sometimes it is intended that the AP authorizes itself to the switch it is attached to and sometimes this is not required, and someone accidentally applies a port profile with authentication enabled.

- Next, check if you see the AP's MAC address appearing as an LLDP neighbor's chassis ID. The port info may be different based on the AP model you have; however, the system name may help you determine more about the state of the AP:
  - When no LLDP system name is reported, the AP has a connection issue. For example, it cannot receive a DHCP lease.
  - When the LLDP system name is "Mist", like in the example below, then the AP is usually up but not assigned to an inventory.

- When the LLDP system name is the MAC address of the AP or the inventory name, then it should already be in the “Connected” state in the inventory of your organization and you can start applying more configuration.

```
root@Switch1> show lldp neighbors
Local Interface    Parent Interface    Chassis Id          Port info          System Name
ge-0/0/1           ae0                 ec:38:73:9a:d5:24   ge-0/0/5           spoke1
ge-0/0/2           ae0                 ec:38:73:9a:d5:24   ge-0/0/6           spoke1
ge-0/0/3           -                   5c:5b:35:be:81:06   ETH0               Mist
```

- Should you have PoE running on the switch, you should also check the power consumption of the AP. The PoE mode negotiated depends on the AP model (usually 802.3af or 802.3at) and can be verified in the datasheet. Depending on the configuration state and radio usage, you should see differences in the actual “power consumed” report.

```
root@Switch1> show poe interface
.
.
root@Switch1> show poe interface ge-0/0/3
PoE interface status:
PoE interface                :  ge-0/0/3
Administrative status        : Enabled
Operational status           :   ON
Power limit on the interface : 19.5W (L)
Priority                     : High
Power consumed               : 7.3W
Class of power device        :        4
PoE Mode                     :   802.3at
  (L) LLDP-negotiated value on the port.
```

- The following checks should be done on the WAN router. In our example, we use a Remote Shell to an SRX Series Firewall acting as a WAN router. You should look for the DHCP lease requests from the AP and verify that you see two sessions using port 443 for TCP/UDP towards the Juniper Mist cloud as shown in the example below:

```
root@spoke1> show dhcp server binding
IP address        Session Id  Hardware address   Expires     State      Interface
10.33.33.12       3           04:5c:6c:6b:13:42  54553       BOUND      ae0.0
10.33.33.15       6           5c:5b:35:be:81:06  48923       BOUND      ae0.0
```

```
root@spoke1> show arp interface ae0.0
MAC Address       Address         Name                      Interface               Flags
04:5c:6c:6b:13:42 10.33.33.12     10.33.33.12               ae0.0                   permanent
5c:5b:35:be:81:06 10.33.33.15     10.33.33.15               ae0.0                   permanent
Total entries: 2
root@spoke1> show security flow session source-prefix 10.33.33.15
Session ID: 34359960425, Policy name: 01_Internet/20, State: Stand-alone, Timeout: 60, Valid
  In: 10.33.33.15/40267 --> 44.204.233.81/443;udp, Conn Tag: 0x0, If: ae0.0, Pkts: 45026,
Bytes: 8408428,
  Out: 44.204.233.81/443 --> 192.168.173.145/23463;udp, Conn Tag: 0x0, If: ge-0/0/0.0, Pkts:
5524, Bytes: 407857,
Session ID: 34359962715, Policy name: 01_Internet/20, State: Stand-alone, Timeout: 1794, Valid
  In: 10.33.33.15/37165 --> 54.144.163.241/443;tcp, Conn Tag: 0x0, If: ae0.0, Pkts: 8997,
Bytes: 4866046,
  Out: 54.144.163.241/443 --> 192.168.173.145/28185;tcp, Conn Tag: 0x0, If: ge-0/0/0.0, Pkts:
5063, Bytes: 421001,
Total sessions: 2
```

NOTE: In rare cases, especially with loaned or lab equipment, someone has already claimed the AP in another organization or cloud and forgotten to release it from inventory. Naturally, this will preclude any usage in another organization. Open a support ticket if that is the case.

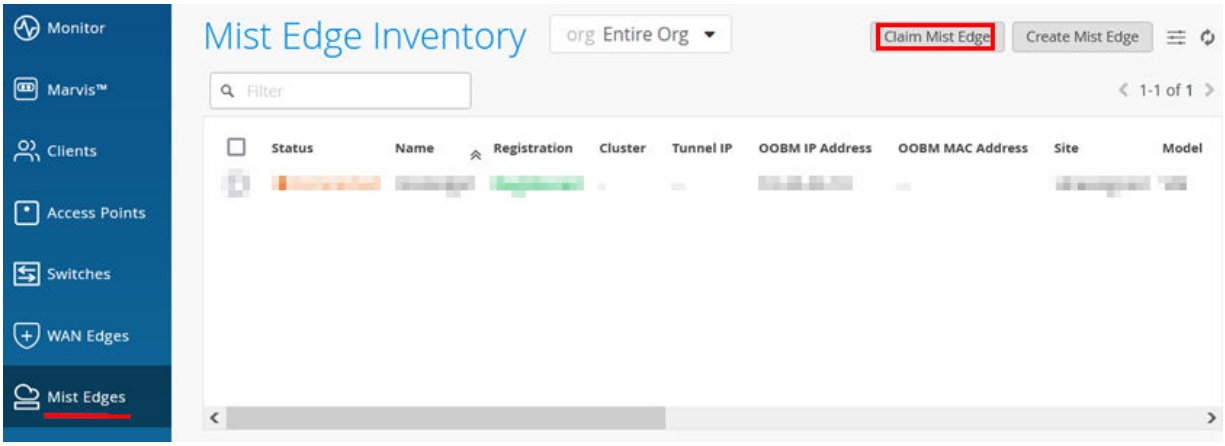
Mist Edge Proxy Installation and Configuration

When deploying Juniper Mist Edge as proxy for RadSec towards the Juniper Mist authentication cloud, the suggested workflow is described in the following steps:

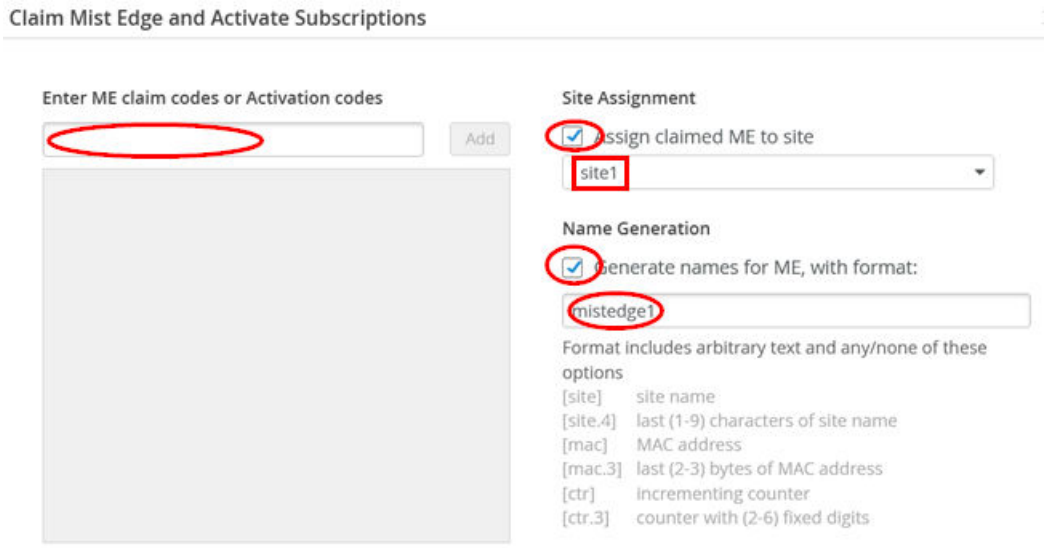
- Deploy the Juniper Mist Edge appliance at the central location where a legacy RADIUS server is usually located. In our simple JVD test lab, it is attached to the WAN router as an exception. In a production environment, the location is usually the corporate headquarters, making the system reachable through the corporate VPN by any branch sites.
- If you only intend to use the Juniper Mist Edge as RADIUS proxy, then you only need to connect the out-of-band management (OOBM) interface. It's used for:
  - Allowing the Juniper Mist Edge to contact the Juniper Mist cloud to get managed.
  - Create the RadSec proxy instance that listens on the RADIUS ports and then tunnels the messages through RadSec towards the Juniper Mist authentication cloud.

- OPTIONAL: Connect the Juniper Mist Edge device's other Interfaces if it is also used for tunnel termination for APs.
- Initially, the OOBM port asks for a DHCP lease, and you must provide one through your infrastructure.
- Get the claim code or QR code by extending the pull-out tag located on the front of the appliance in the lower-right corner. This allows for ZTP.
- Create the device in the Juniper Mist cloud via the claim code.
- Wait for the Juniper Mist Edge device to appear as "Connected" and "Registered".
- Perform an update.
- Instead of a DHCP lease, configure a static IP address on the OOBM port.

We assume that the Juniper Mist Edge is already connected and powered on in the lab, so we can now create the instance in the Juniper Mist cloud. Go to **Mist Edges > Mist Edge Inventory > Claim Mist Edge**:

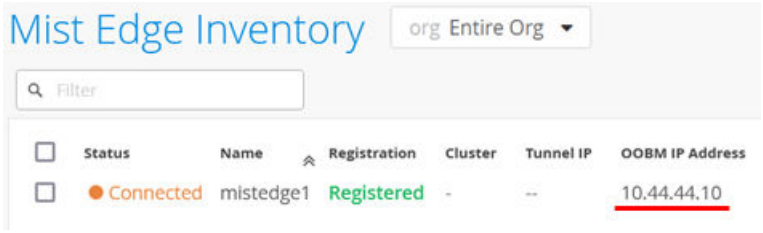


Claim the new Mist Edge like the example shown below:

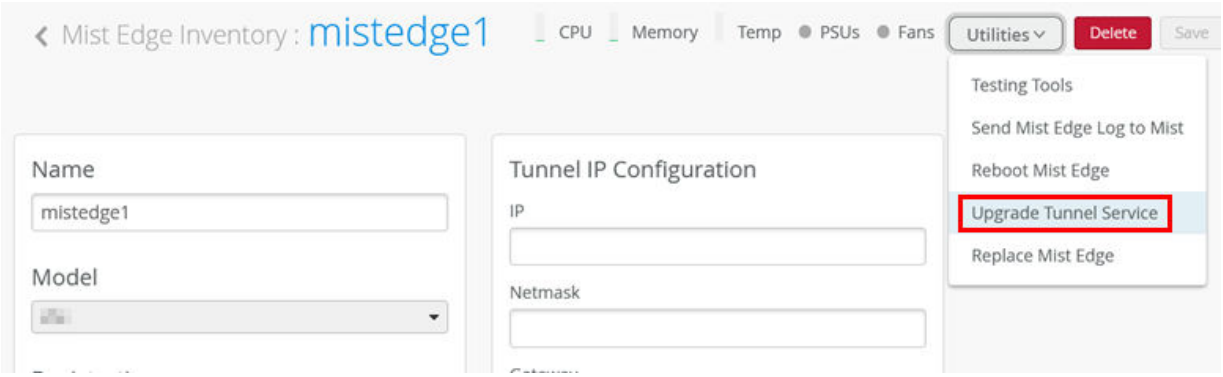


Remember that the claim code is located in the right front bottom of the appliance, and you must extend the pull-out tag to view it:

Next, the Juniper Mist Edge should appear as “Connected” and “Registered” in the Juniper Mist Edge inventory and should show the OOBM interface’s DHCP-assigned IP address.

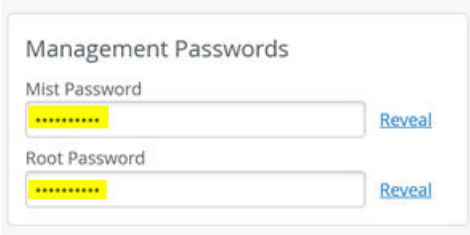


After clicking on the device in the portal, a new page opens. The first task (which may not be needed) is to perform an update as shown below:



.

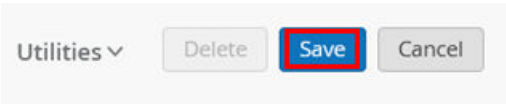
Next, we recommend changing the default management password for the internal Mist and root accounts on the appliance.



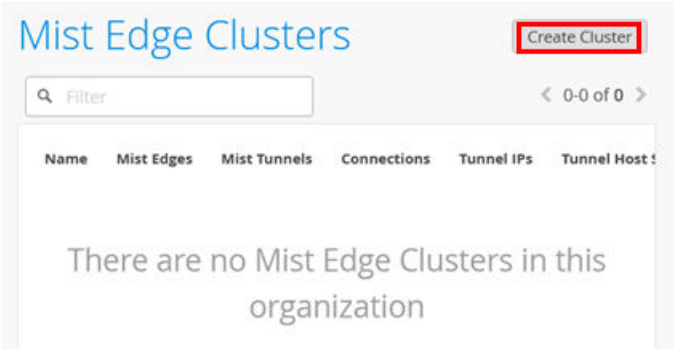
Next, we need to change the dynamic IP address assigned on the OOBM interface to a static IP address since all RADIUS clients expect to reach the RADIUS server using a static IP address. Hence, we configure for our lab:

- Configure static OOBM IP=`Enabled`
- IP Address=`10.44.44.5`
- Subnet Mask=`255.255.255.0`
- Default Gateway=`10.44.44.1` (our WAN router's ge-0/0/5 interface)
- DNS=`8.8.8.8, 1.1.1.1`

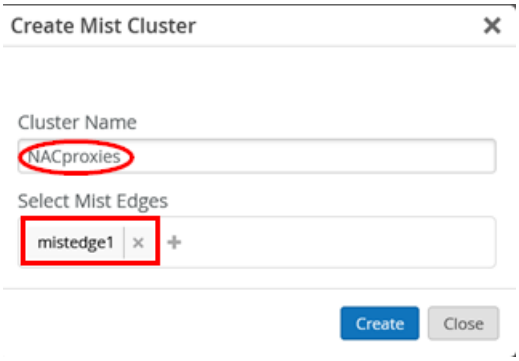
Do not forget to save your new configuration.



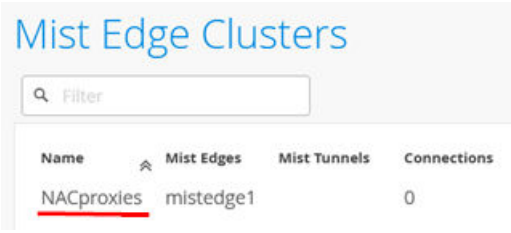
Next, go back one page and select **Create Cluster**.



Name the cluster and select the Mist Edge to be a part of it.

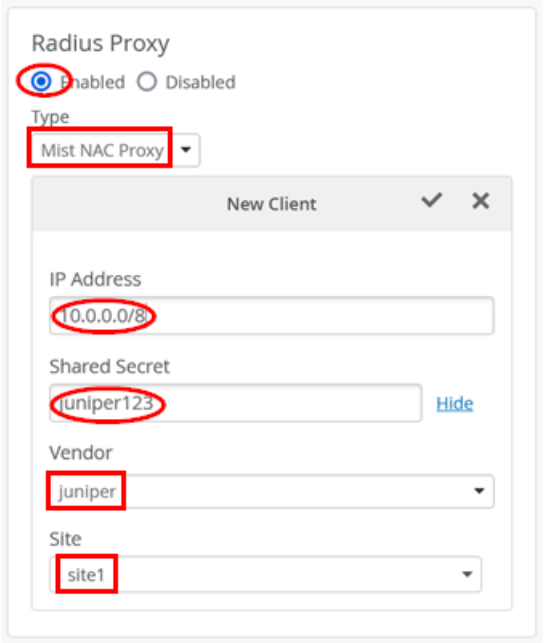


Select the Cluster:

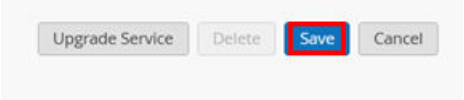


Next, enable the RadSec proxy like in our example:

- Radius Proxy=`Enabled`
- Type=`Mist NAC Proxy`
- IP Address=`10.0.0.0/8` (We should manage all IP addresses one by one, but not for this lab).
- Shared Secret=`juniper123` (We should use individual shared secrets, but this is a lab).
- Vendor=`juniper` (In this case, "Juniper," but can be a third-party vendor selected from the drop-down menu).
- Site=`site1`



Do not forget to save your configuration changes.



(Optional) If you prefer, you can open an SSH shell as root to the Juniper Mist Edge device and check locally:

- ```
  root@mistedge1:~# ss -tuln
  Netid  State   Recv-Q  Send-Q   Local Address:Port   Peer Address:Port Process
  udp    UNCONN  0       0              0.0.0.0:1812        0.0.0.0:*
  udp    UNCONN  0       0              0.0.0.0:5355        0.0.0.0:*
  udp    UNCONN  0       0        127.0.0.53%lo:53          0.0.0.0:*
  udp    UNCONN  0       0              0.0.0.0:68          0.0.0.0:*
  udp    UNCONN  0       0                 [::]:1812           [::]:*
  udp    UNCONN  0       0                 [::]:5355           [::]:*
  tcp    LISTEN  0       128            0.0.0.0:22          0.0.0.0:*
  tcp    LISTEN  0       4096         127.0.0.1:9080        0.0.0.0:*
  tcp    LISTEN  0       4096           0.0.0.0:5355        0.0.0.0:*
  tcp    LISTEN  0       4096         127.0.0.1:9199        0.0.0.0:*
  tcp    LISTEN  0       4096         127.0.0.1:9109        0.0.0.0:*
  tcp    LISTEN  0       4096     127.0.0.53%lo:53          0.0.0.0:*
  tcp    LISTEN  0       128               [::]:22             [::]:*
  tcp    LISTEN  0       4096              [::]:5355           [::]:*
  root@mistedge1:~# systemctl status radsecproxy
  * radsecproxy.service - radsecproxy
       Loaded: loaded (/lib/systemd/system/radsecproxy.service; enabled; vendor p>
      Drop-In: /usr/lib/systemd/system/radsecproxy.service.d
               `-mxedge.conf
       Active: active (running) since Thu 2024-08-15 15:27:29 UTC; 15min ago
         Docs: man:radsecproxy(1)
     Main PID: 1821 (radsecproxy)
           IP: 189.8K in, 128.8K out
           IO: 128.0K read, 72.0K written
        Tasks: 9 (limit: 32768)
       Memory: 1.6M
          CPU: 592ms
       CGroup: /system.slice/radsecproxy.service
               `-1821 /usr/sbin/radsecproxy -c /etc/mxedge-radsecproxy.conf
  Aug 15 15:27:29 mistedge1 systemd[1]: Starting radsecproxy...
  Aug 15 15:27:29 mistedge1 systemd[1]: Started radsecproxy.
  ```

NOTE: The RadSec tunnel is always created on demand and does not remain permanently open.
  ```

Juniper Mist Authentication Cloud Certificate Installation

In this section, we demonstrate installing the certificates needed for the various authentication methods. In Table 3 on page 114, we highlight the needed certificates per authentication methods and where they need to be installed. Remember that usually the customer PKI is used for all EAP methods where certificates come into play. For the last two methods in Table 3 on page 114, we reuse the automatic PKI Mist creates for each organization internally for the RadSec tunnels. This makes life easier in cases when deploying certificates rather than getting them from the customer's PKI.

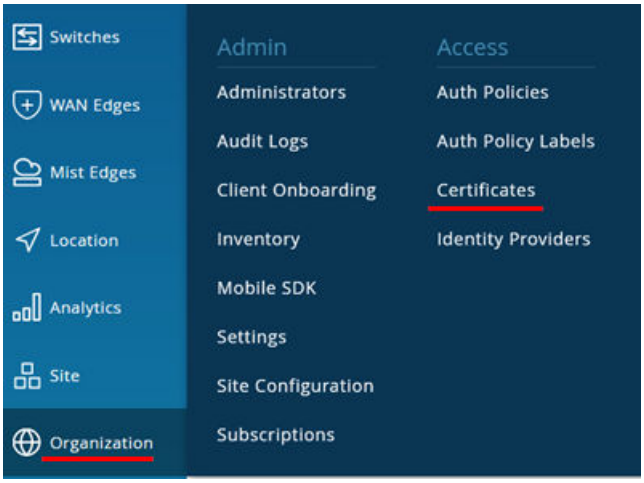
NOTE: In the below table, the term "CA-cert" refers to when the customer has multiple levels with intermediate or signing CAs in their PKI. All of these public certificates must be installed for the entire path. The system needs to be able to evaluate the entire path back to the root CA.

Table 3: Certificate Installation Requirements

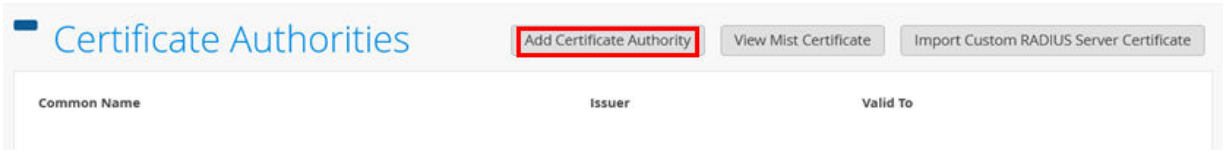
Authentication Method	CA-Certificate in Mist	RADIUS Server in Mist	Client Supplicant CA-Cert	Client Supplicant User/Machine Cert
MAB	N/A	N/A	N/A	N/A
EAP-TLS	Customer PKI CA-cert	Customer PKI TLS-Server Public/Private	Customer PKI CA-cert	Customer PKI TLS-Client Public/Private
EAP-TTLS	Customer PKI CA-cert	Customer PKI TLS-Server Public/Private	Customer PKI CA-cert	N/A
EAP-TTLS Mist PKI	Mist PKI CA-cert	Mist PKI internal	Mist PKI CA-cert	N/A
EAP-TLS for Mist AP	Mist PKI CA-cert	Mist PKI internal	Mist PKI CA-cert	Mist PKI auto deployed on AP

Import a Customer CA Certificate

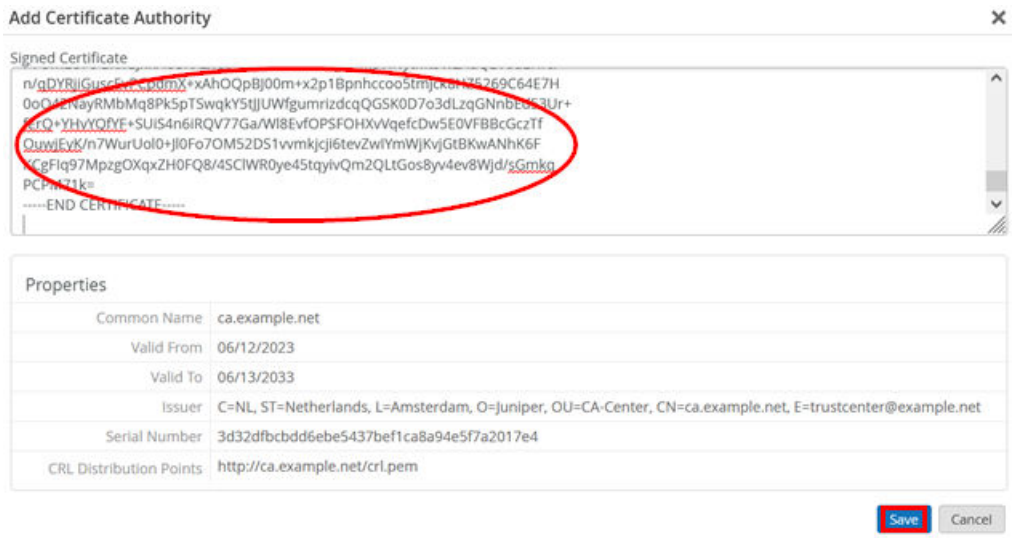
Go to **Organization > Certificates** to start all processes in this chapter.



You should see no certificate installed yet, so let's import the root CA from the customer's PKI. You need the public part of the certificate in PEM format, then click on **Add Certificate Authority**:

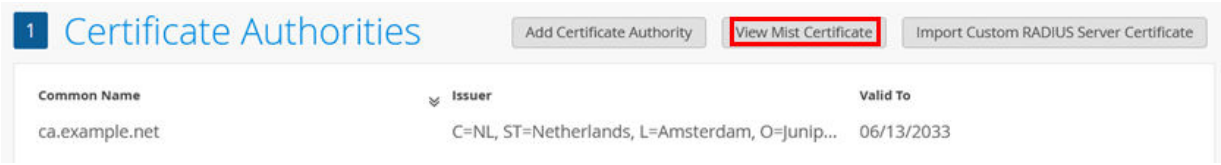


Paste the PEM part of the certificate into the **Signed Certificate** field. Then, check the extracted properties and then click **Save**.

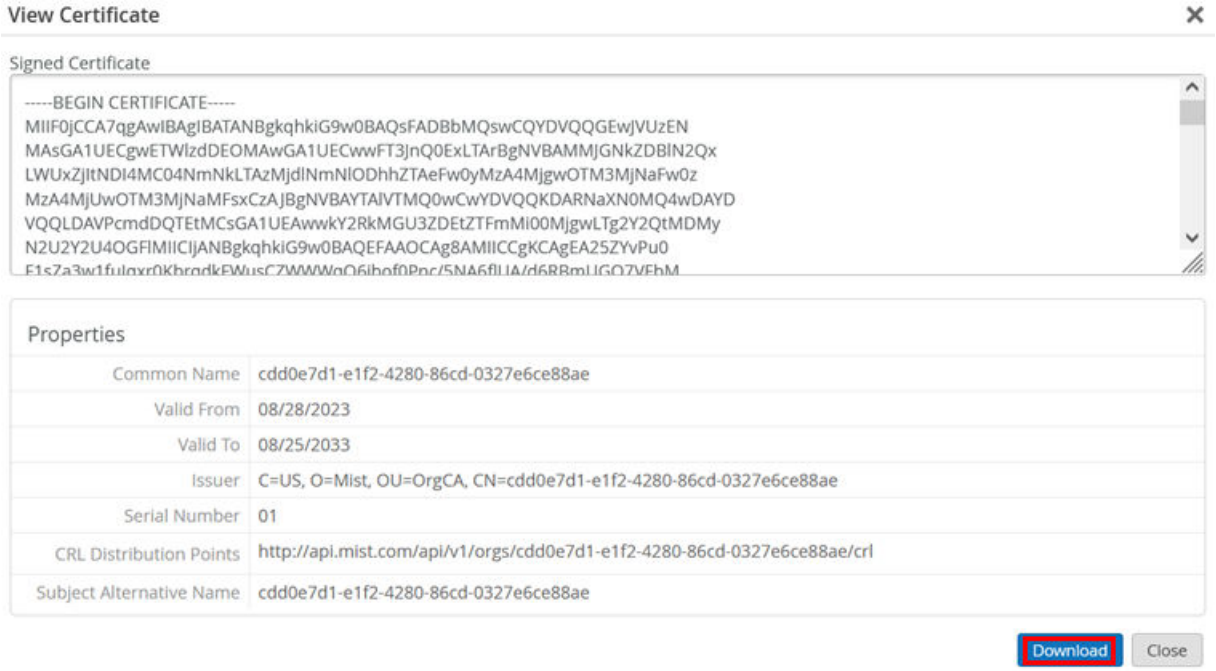


Review and Import the Mist CA Certificate

We continue in the same dialogue window and now click on **View Mist Certificate**:

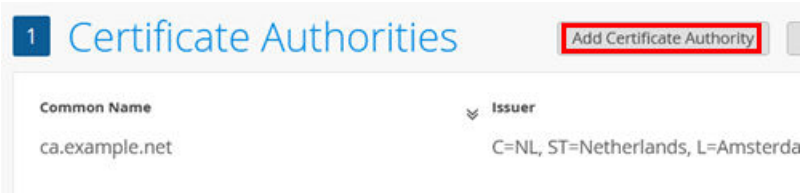


Inspect the certificate and click on **Download** to save the certificate locally.

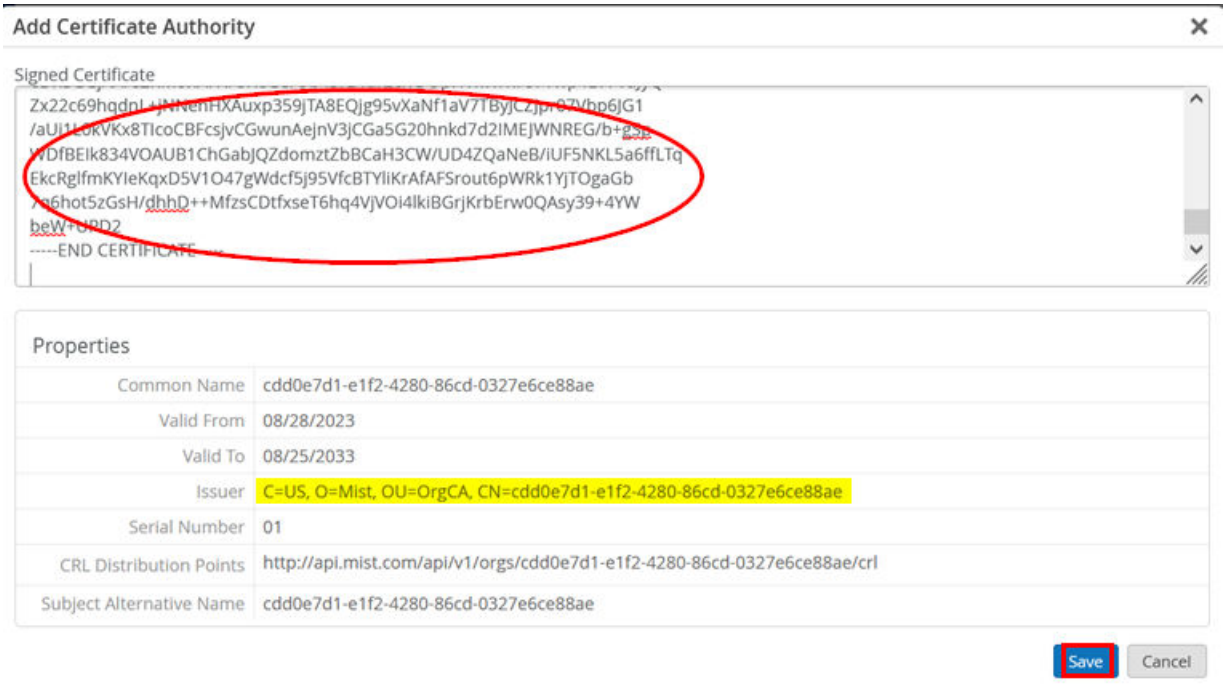


Open the downloaded file and copy the entire contents:

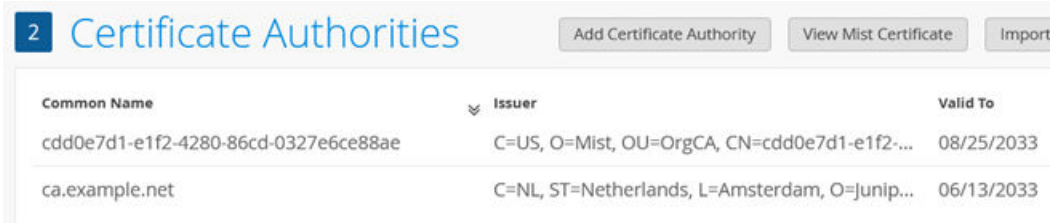
Again, click on **Add Certificate Authority**:



Paste the downloaded contents of the certificate into the **Signed Certificate** field and click **Save**:



You should now have at least two CA certificates, similar to that shown in the example below:

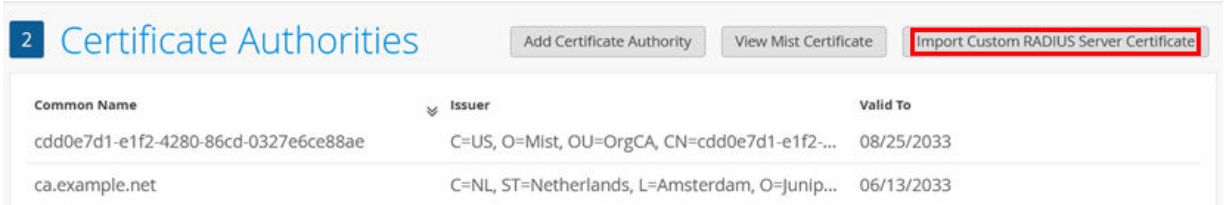


Import the RADIUS Server Certificates

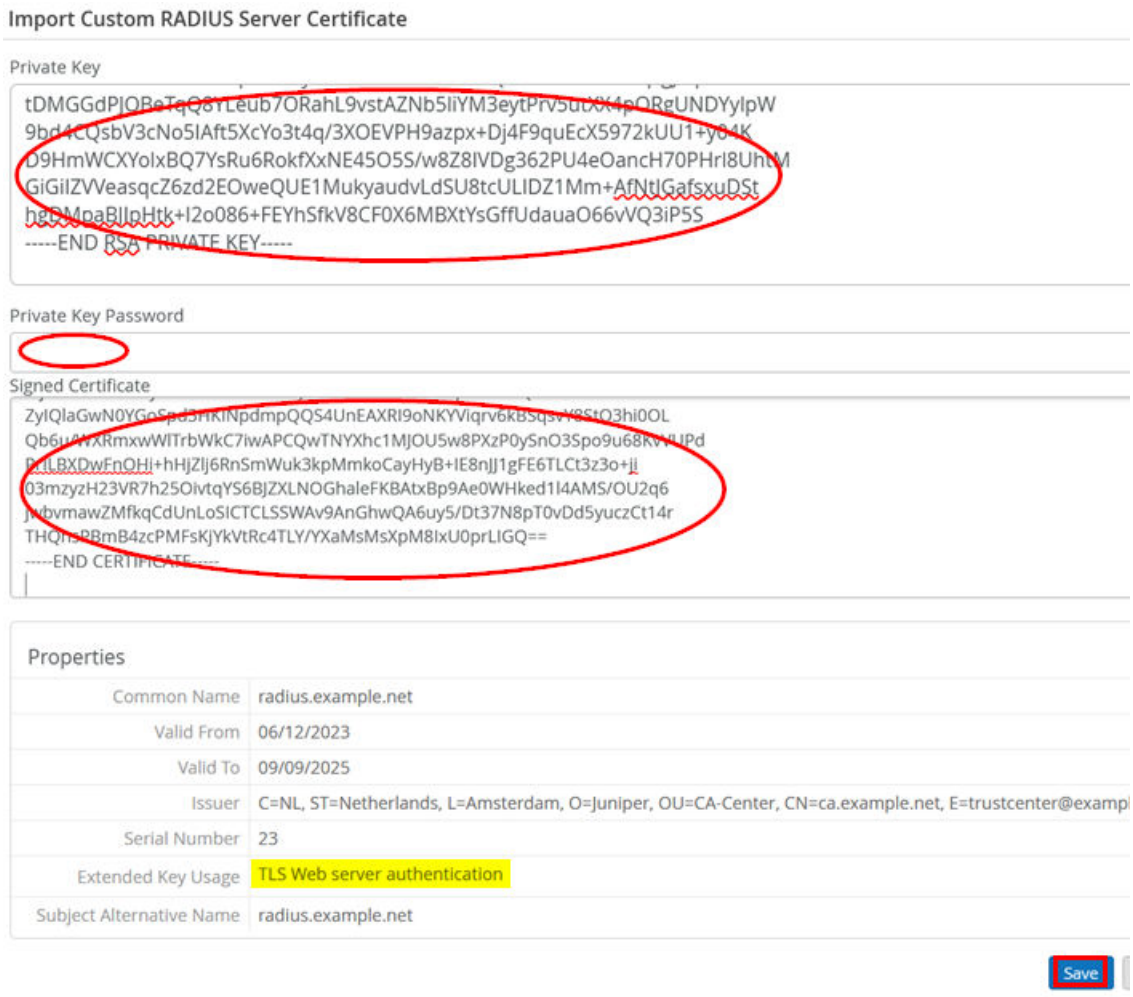
We continue in the same dialogue and check the status of the RADIUS server certificate:

- If the field states **Import Customer RADIUS Server Certificate,** then nothing is defined yet and the RADIUS server will use a certificate signed by the Mist internal PKI for each organization.
- If the field states **View Customer RADIUS Server Certificate,** then you have valid customer PKI certificates successfully loaded, which will be used for EAP authentications.

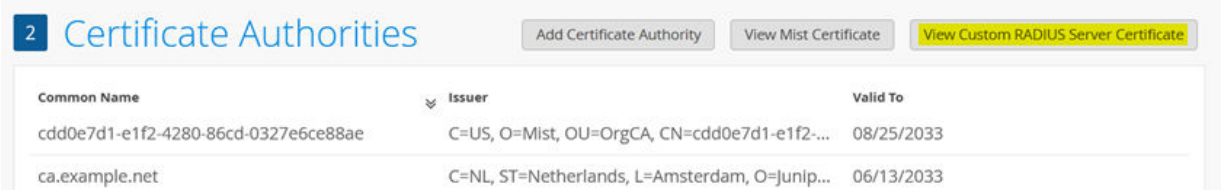
In our example, nothing is defined yet, hence we click on the **Import Customer RADIUS Server Certificate** button.



Fill in the fields for **Private Key** and **Signed Certificate**. You may also need to add a password under **Private Key Password**. Once finished, check the information displayed and the properties before you click **Save**.



Now all required certificate configurations for your lab have been performed.



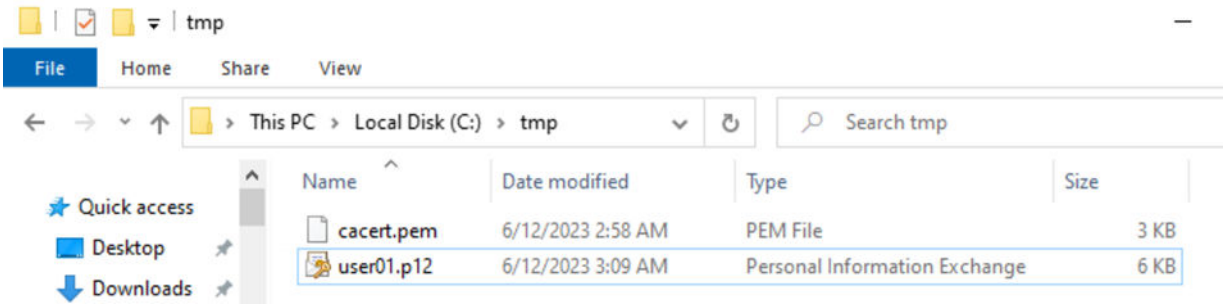
Configure Client Supplicants with Certificates and Necessary EAP Methods

This chapter has examples of manual client configurations used when no MDM is set up for managing supplicants. This is intended more for test engineers rather than for production-grade deployments where using an MDM is highly recommended. However, it may be good to know what needs to be configured and where.

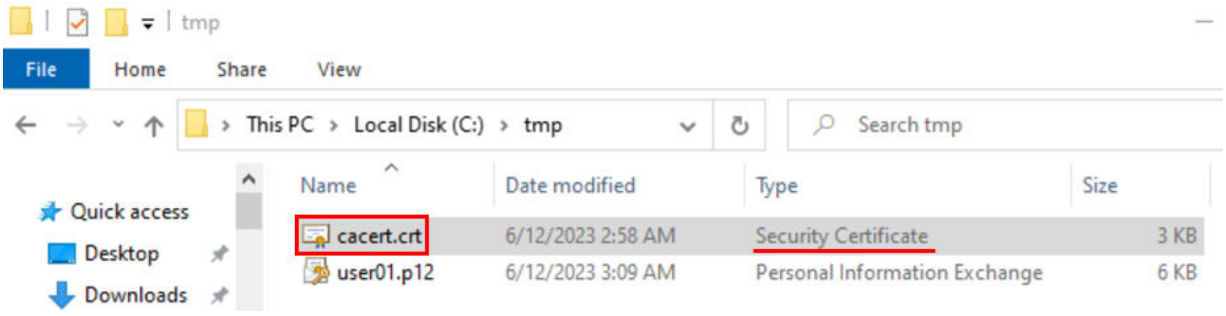
> **NOTE**: Again, if you intend to use an MDM of some kind to manage your clients you can skip over this entire chapter.

Installing a Local Certificate on a Windows Client

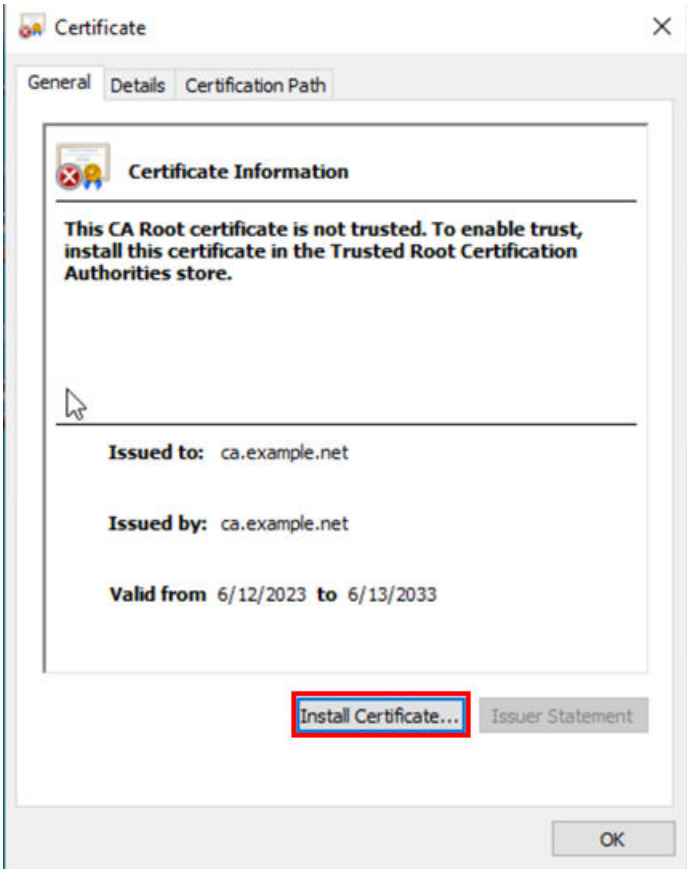
In our example, we use a home-grown certificate PKI. Here, you can see the folder where the original `cacert.pem` and `user01.p12` files are located and transferred to the local device:



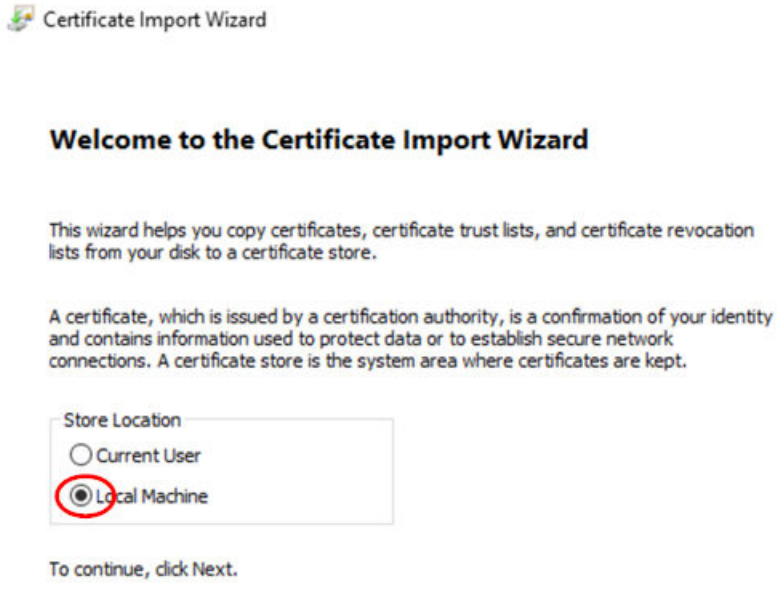
Rename the file `cacert.pem` to `cacert.crt` so that Windows recognizes the correct type. Then, double-click on the file to import it.



Click on **Install Certificate.**



Select **Local Machine** to be able to share the new root CA with other accounts.

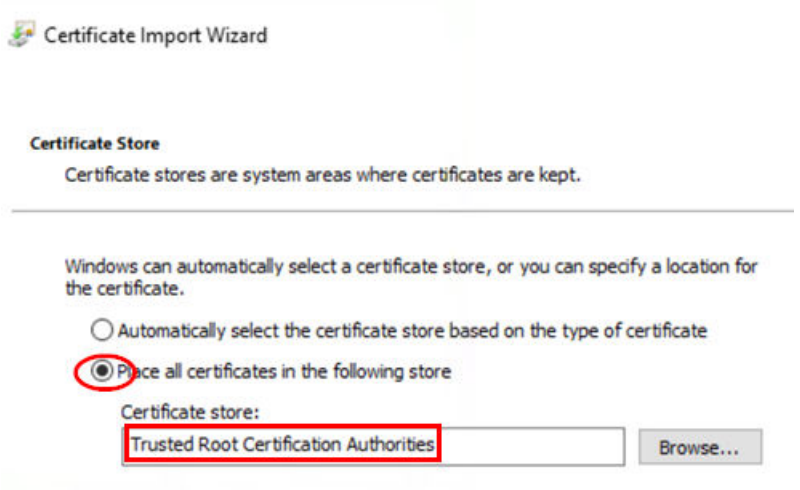


Configure the following settings:

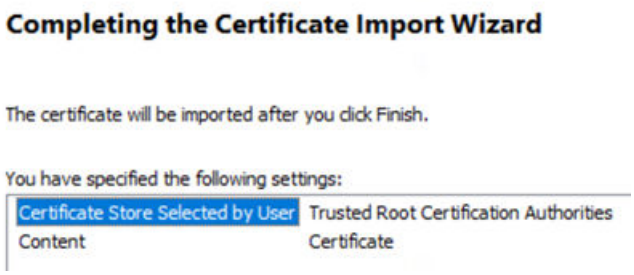
- Place all certificates in the following store=`Checked`

- Certificate store=

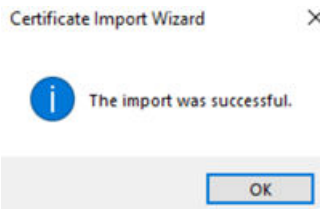
```
Trusted Root Certification
Authorities
```



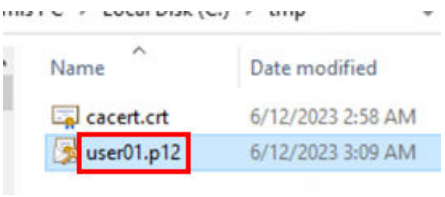
Complete the wizard:



Your certificate should now be imported.

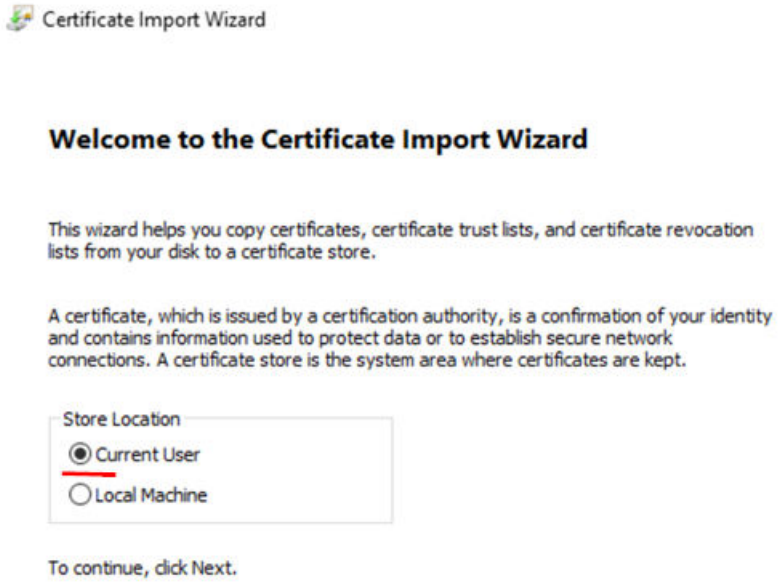


Next, let's import the User=Certificate by double-clicking the file `user01.p12`:

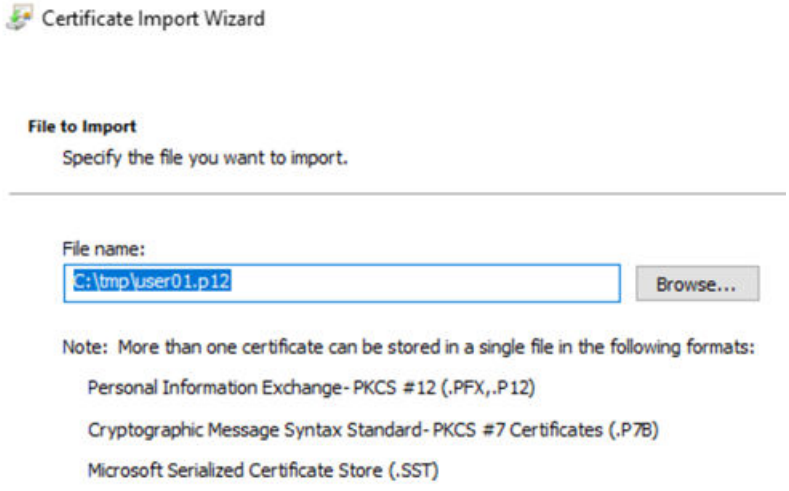


Configure in this dialogue:

- Storage Location=`Current User`

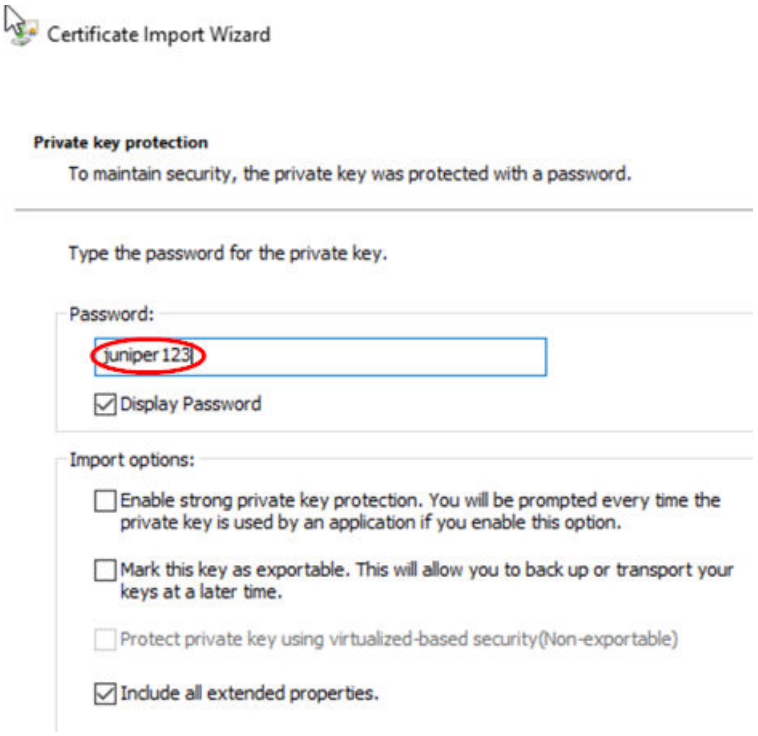


Review the file location and name:



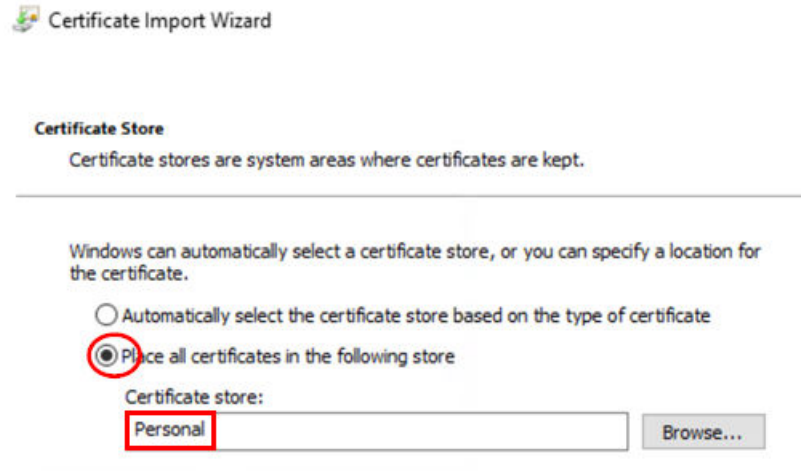
Our certificate has the following information:

- Password=juniper123

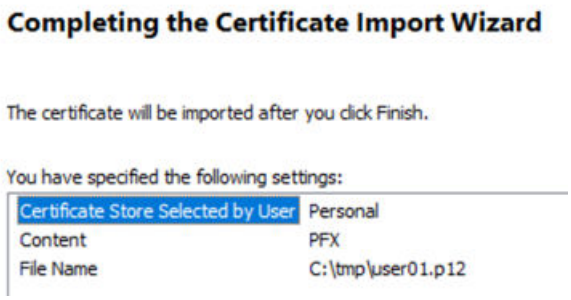


Configure the following settings:

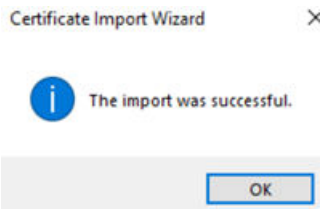
- Place all certificates in the following store=Checked
- Certificate store=Personal



Complete the wizard.

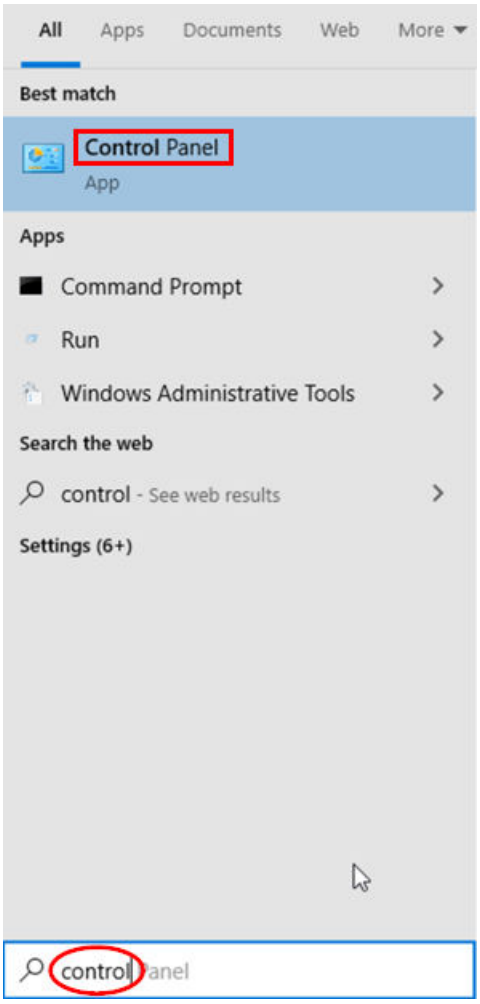


Your certificate should now be imported.



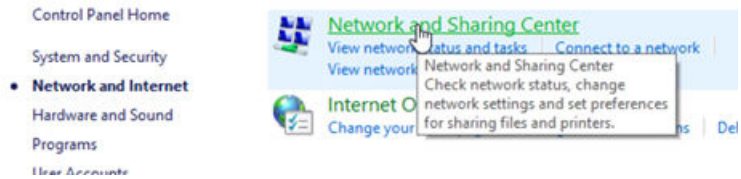
Windows Client Wired EAP-TLS Example

To configure the wired NIC, type the word “control” in the local search field and click **Control Panel**:

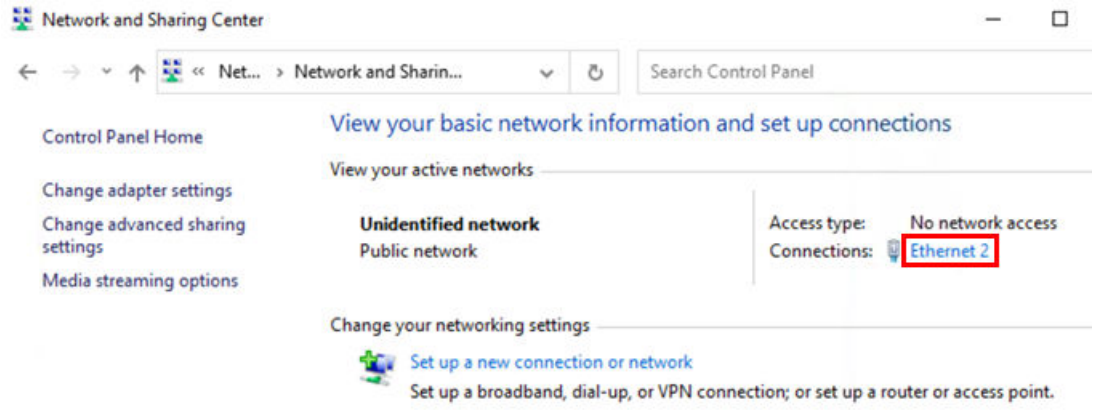


Depending on your view, select **Network and Internet**:

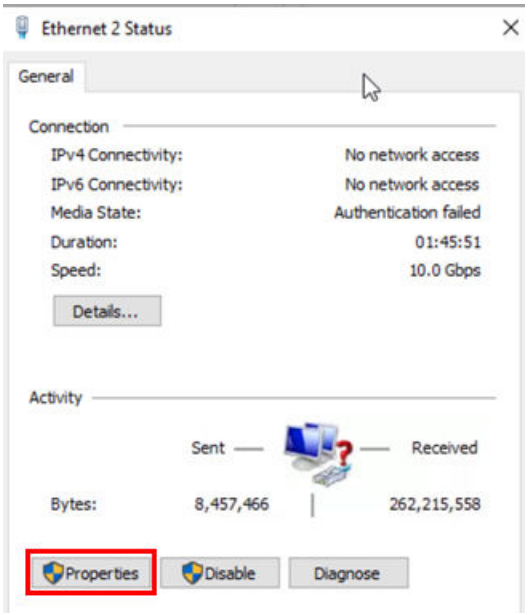
Depending on your view, select **Network and Sharing Center**.



There should be at least one ethernet adapter listed under **Connections** that you can select.



Select the **Properties** dialogue.

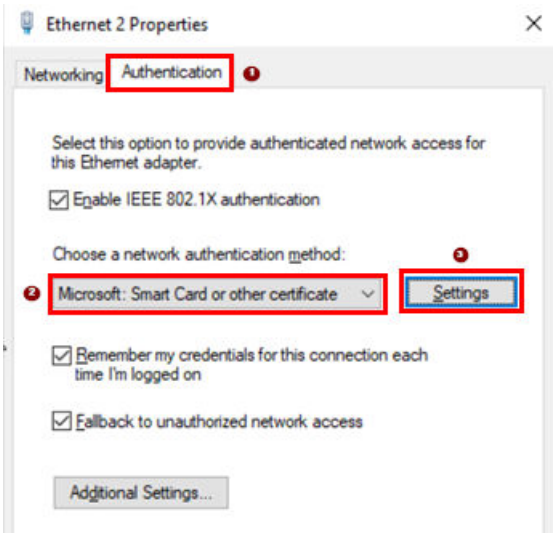


Change to the **Authentication** tab. Then, configure the following settings:

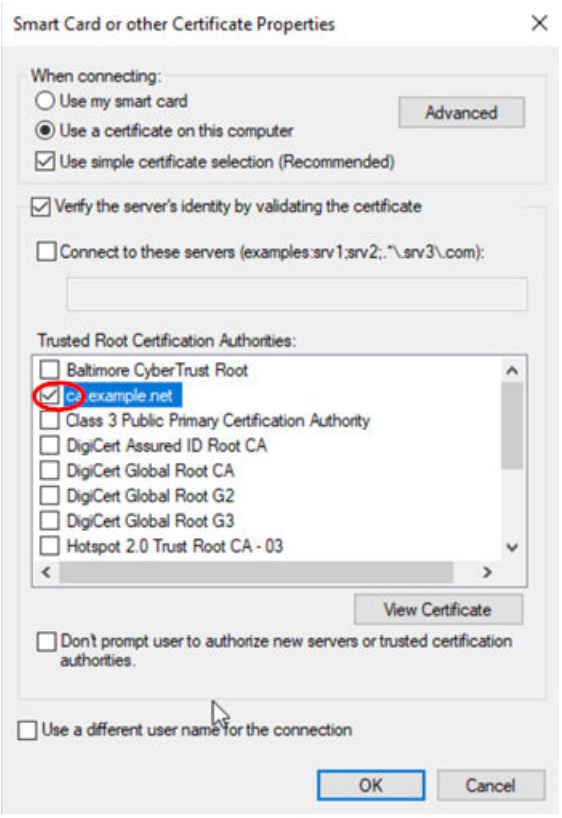
- Network authentication method=

```
Microsoft: Smart Card or
other certificate
```

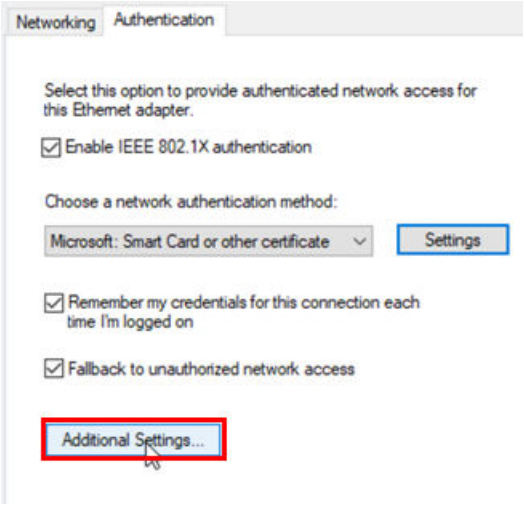
- Click on=`Settings`



The only change in this dialogue is to select the **Trusted Root Certification Authorities** which in our case is `ca.example.net`. Leave the remaining fields as their defaults and return to the previous dialogue.

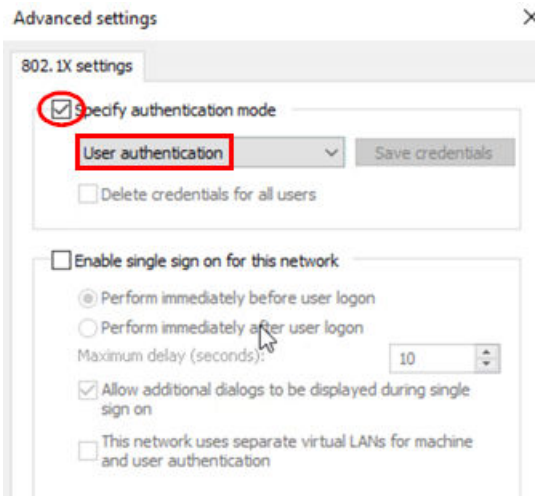


Click on **Additional Settings**.

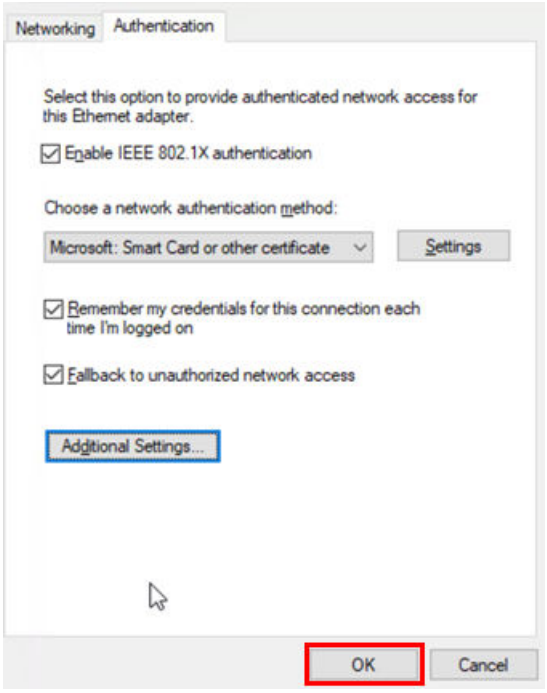


Configure the following settings:

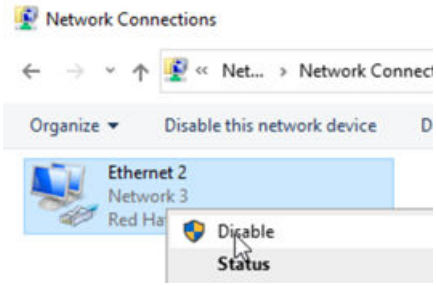
- Specify authentication mode=`Checked`
- Mode=`User authentication`



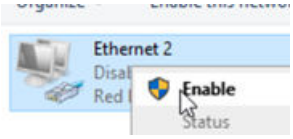
Now, you can finish the main dialogue.



For simple testing, we recommend you first disable the adapter.

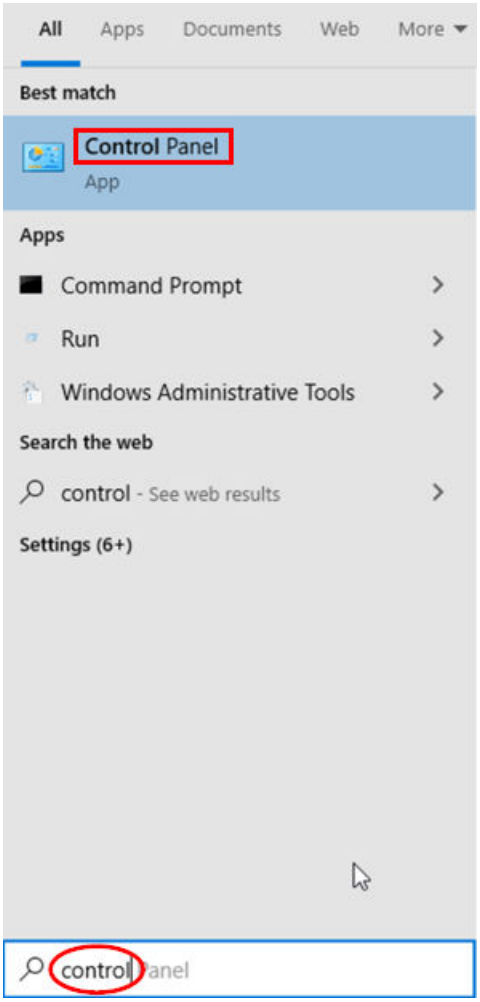


Then, enable the adapter again and check if authentication works.

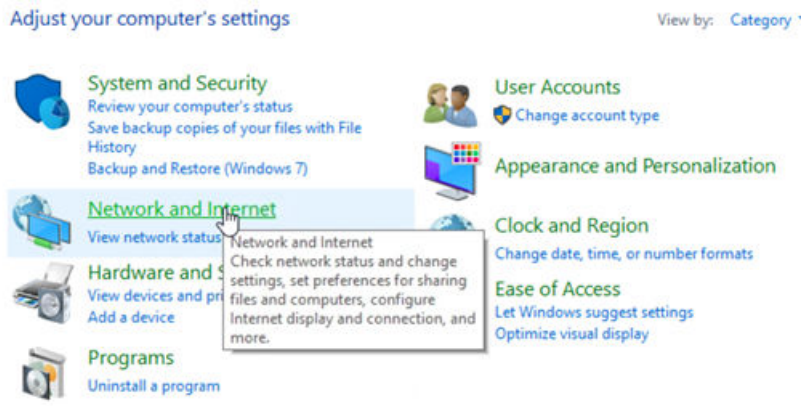


Windows Client Wireless EAP-TLS Example

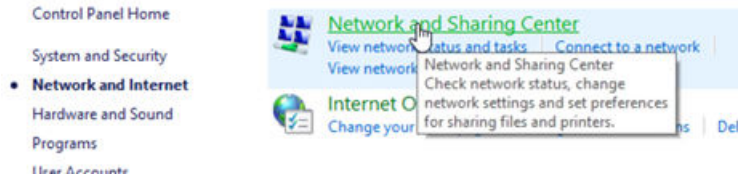
To configure the WLAN adapter, type the word “control” in the local search field and click **Control Panel.**



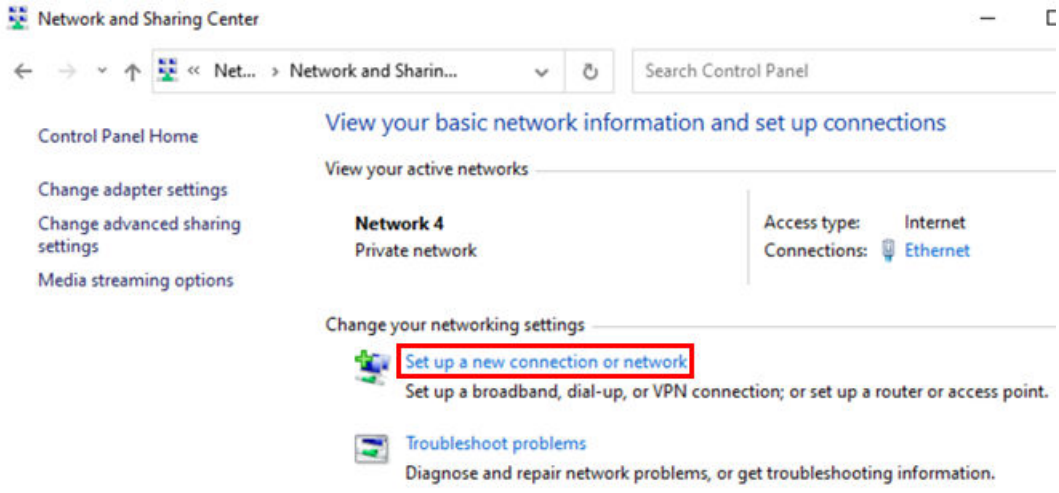
Depending on your view, select **Network and Internet**.



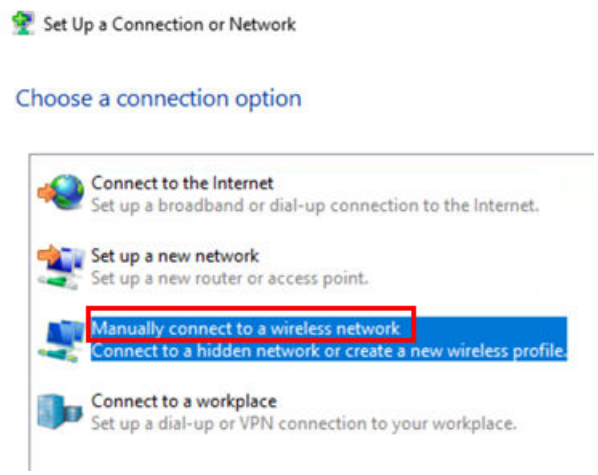
Depending on your view, select **Network and Sharing Center**.



Then, select **Set up a new connection or network**.



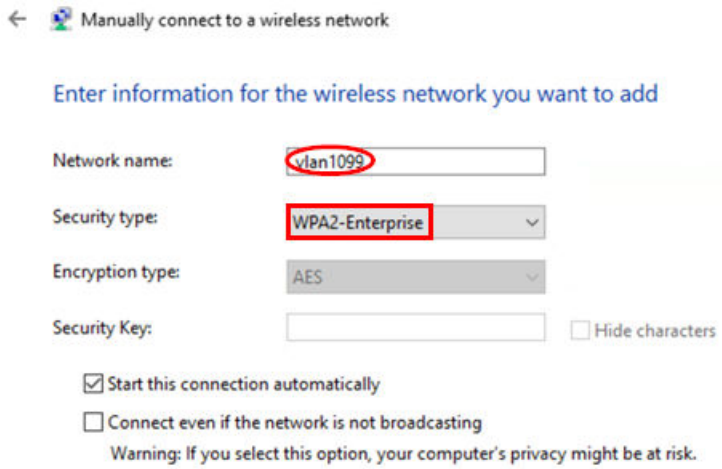
Select **Manually connect to a wireless network**.



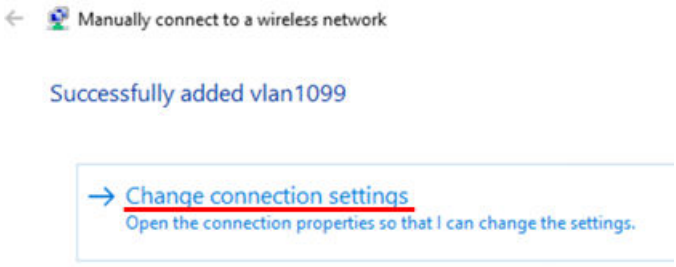
Configure the following settings:

- Network Name=`<SSID>`
- Security type=`WPA2-Enterprise`

.



Select **Change connection settings**.

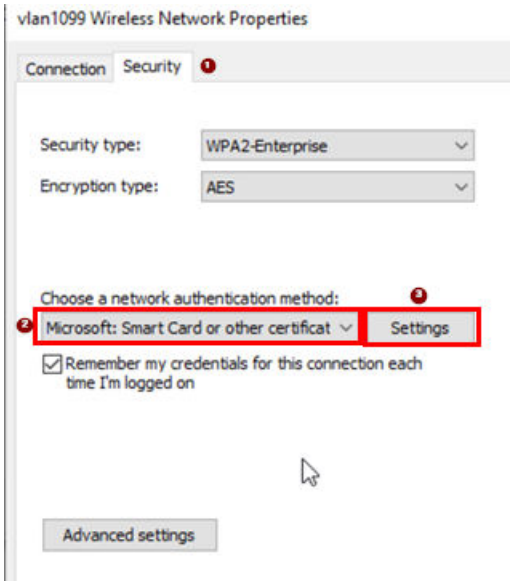


Change to the **Security** tab. Then, configure the following:

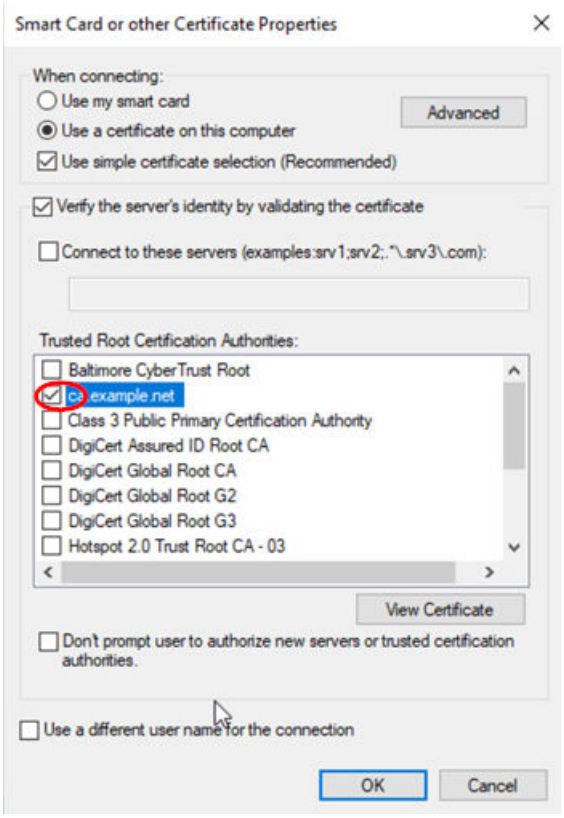
- Network authentication method=

```
Microsoft: Smart Card or
other certificate
```

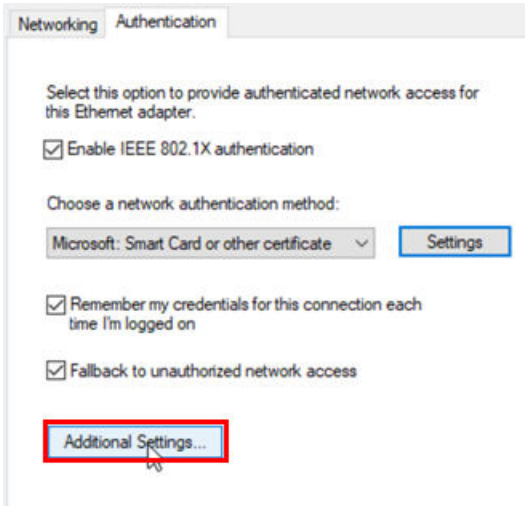
- Click on=`Settings`



The only change in this dialogue is to select the **Trusted Root Certification Authorities** which in our case is `ca.example.net`. Leave the remaining fields as their defaults and return to the previous dialogue.

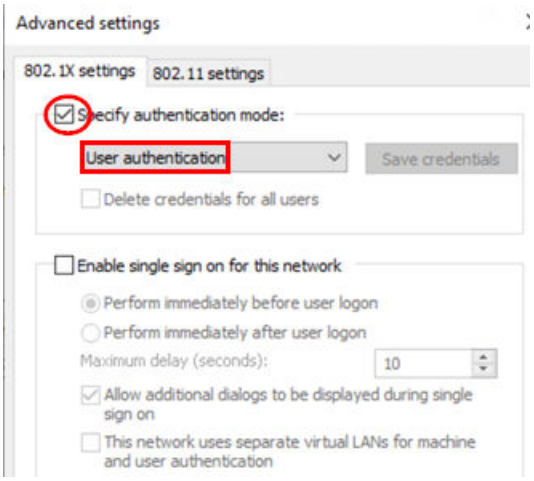


Click on **Additional Settings**.

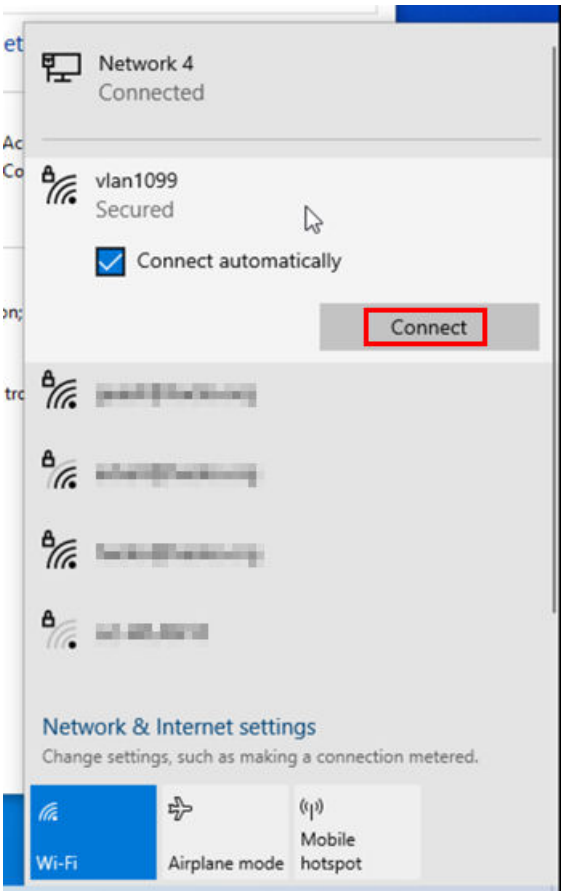


Configure the following settings.

- Specify authentication mode=`Checked`
- Mode=`User authentication`

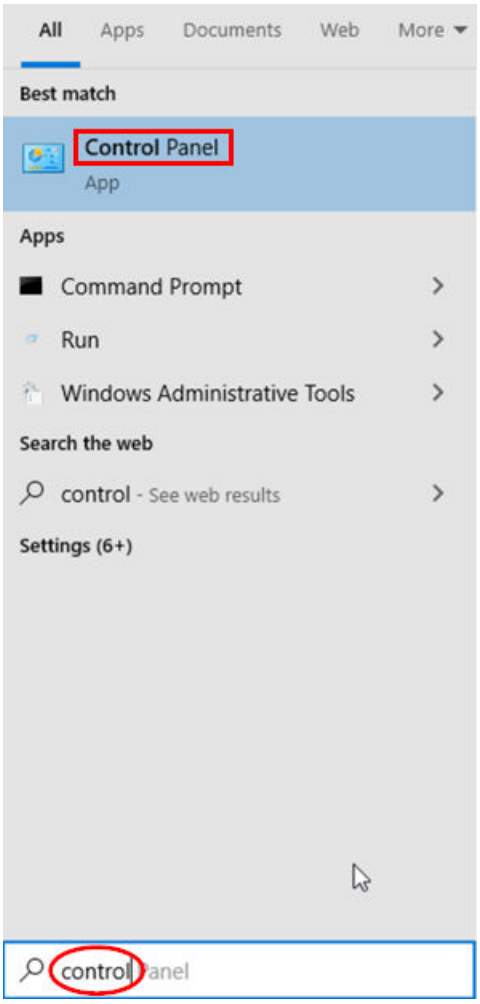


You can now finish the main dialogue. Then, connect to the wireless network using the usual dialogue.



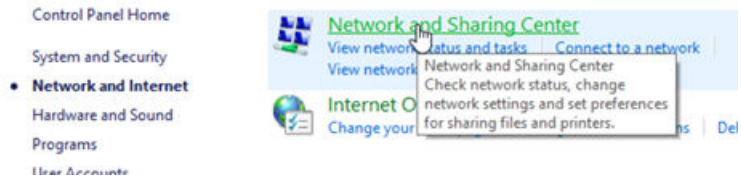
Windows Client Wired EAP-TTLS Example

To configure the wired NIC, type the word “control” in the local search field and click **Control Panel**.

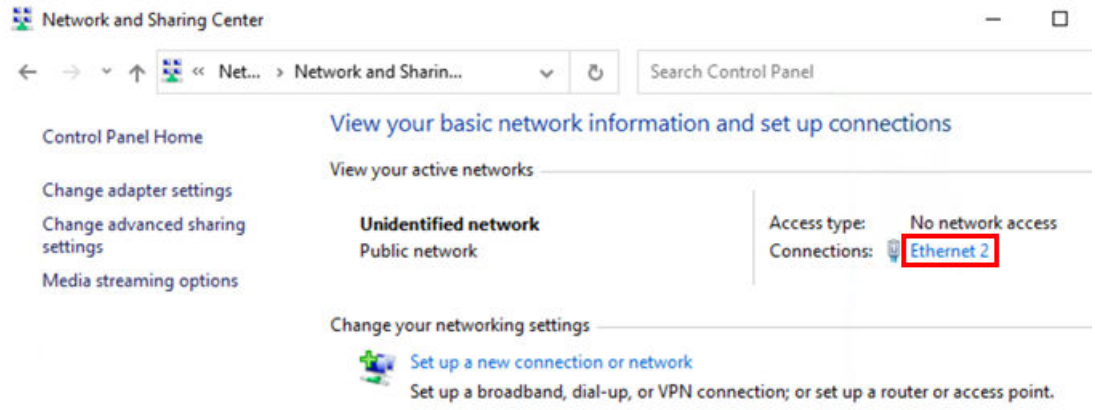


Depending on your view, select **Network and Internet**.

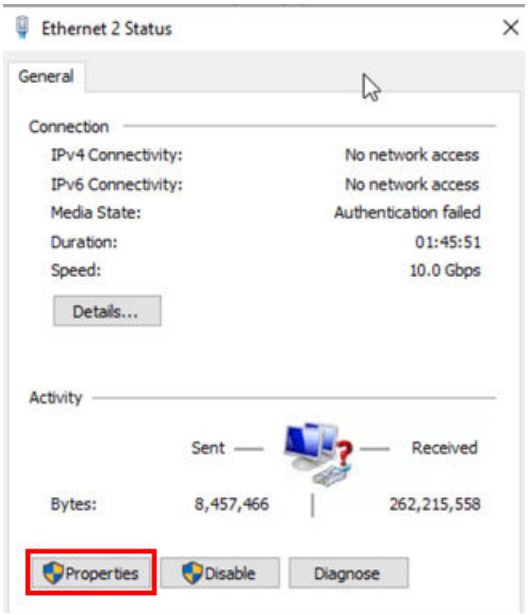
Depending on your view, select **Network and Sharing Center**.



There should be at least one ethernet adapter under **Connections** that you can select.

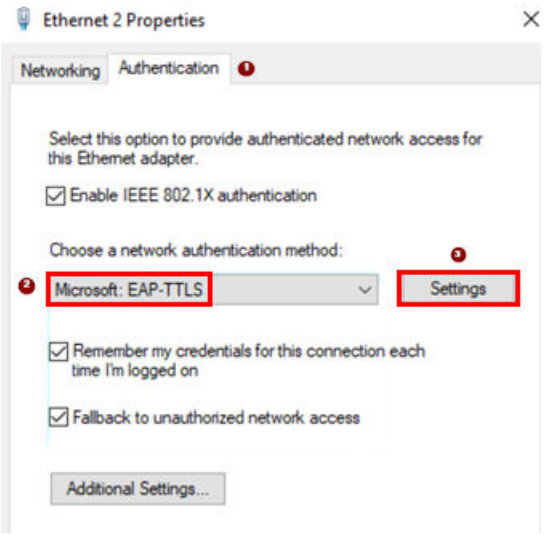


Select the **Properties** dialogue.

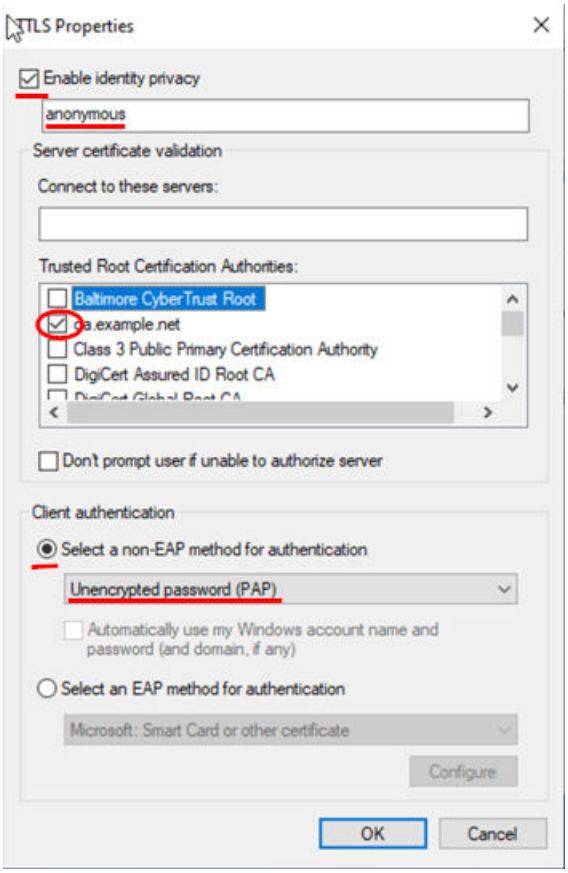


Change to the **Authentication** tab. Then, configure the following settings:

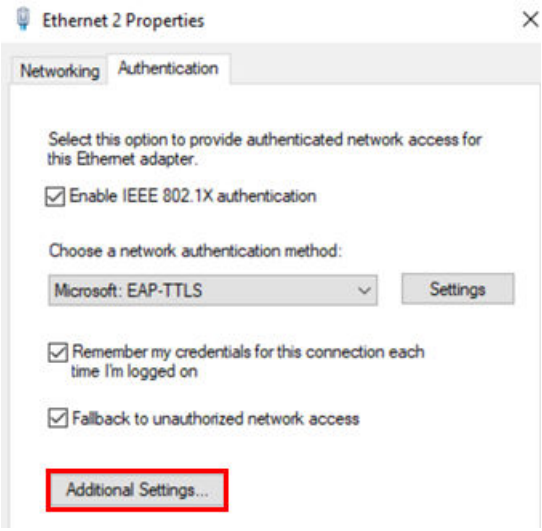
- Network authentication method=Microsoft: `EAP-TTLS`
- Click on=`Settings`



The only change in this dialogue is to select our **Trusted Root Certification Authorities,** which in our case is `ca.example.net`. Leave the remaining fields as their defaults and return to the previous dialogue. Make sure the client authentication method is the default **Unencrypted password (PAP)**.

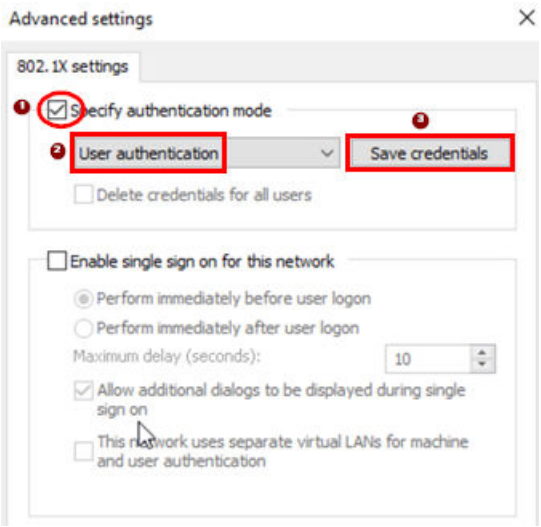


Click on **Additional Settings**.



Configure the following settings:

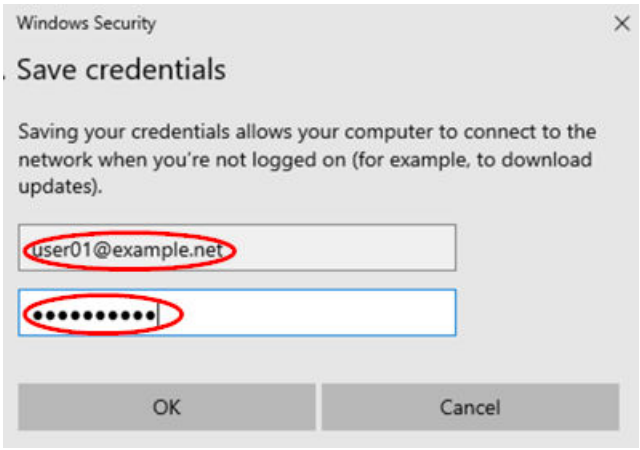
- Specify authentication mode=`Checked`
- Mode=`User authentication`
- Click-on=`Save credentials`



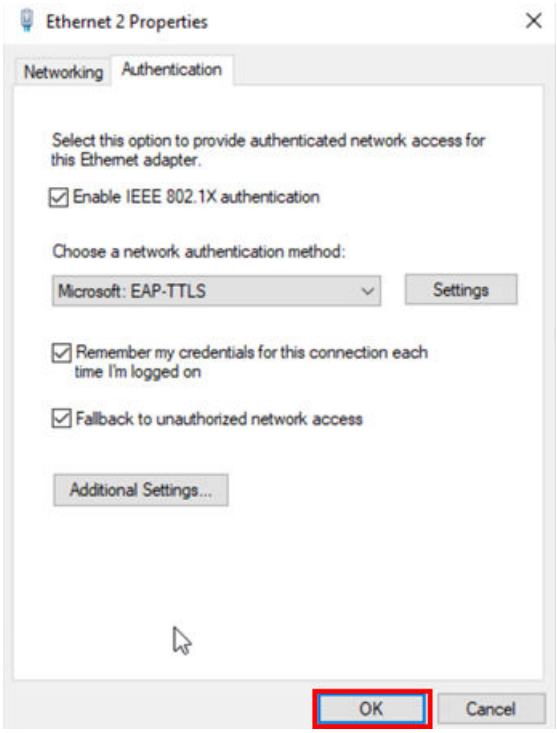
Next, enter a username and password that is valid for the remote IdP the Juniper Mist Authentication cloud will contact for validation. In our case, with the example PKI, enter the following:

- Username=`user01@example.net`
- Password=`juniper123`

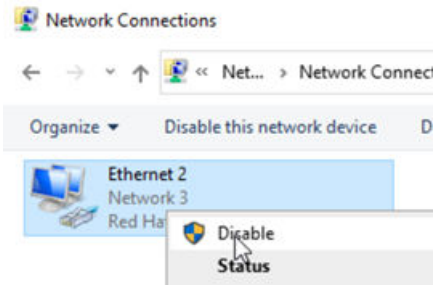
Then, save this dialogue and return to the previous one.



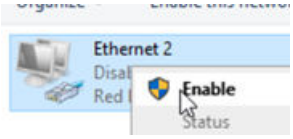
You can now finish the main dialogue.



For simple testing, we recommend you first disable the adapter.

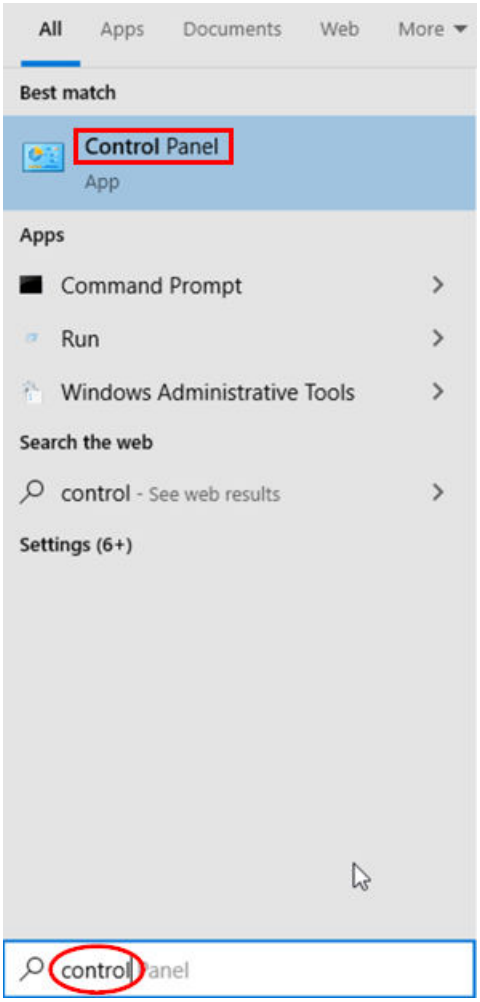


Then, enable the adapter back again and check if authentication works.

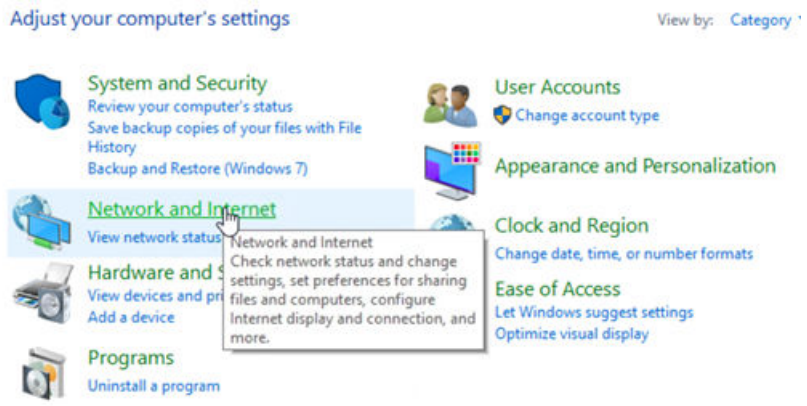


Windows Client Wireless EAP-TTLS Example

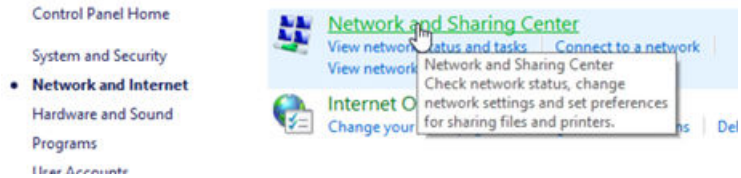
To configure the WLAN adapter, type the word “control” in the local search field and click **Control Panel**.



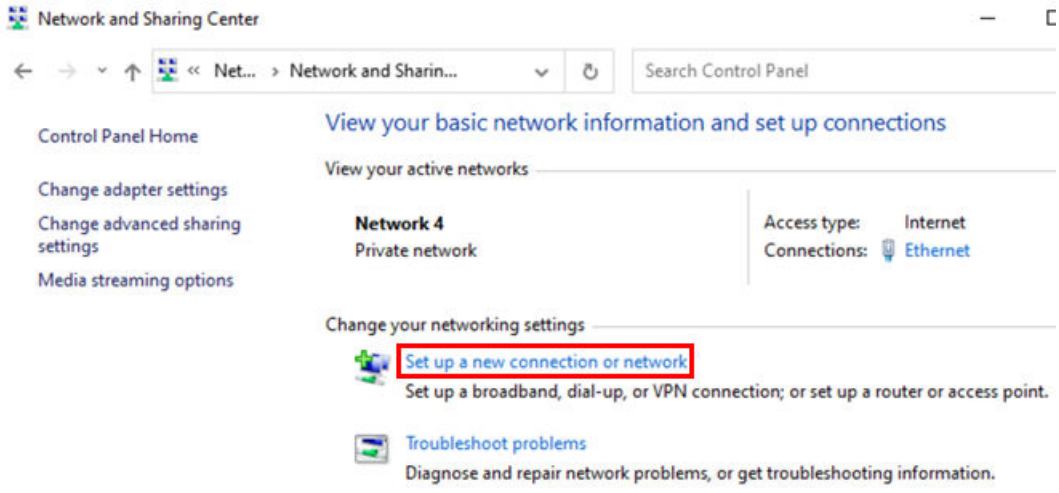
Depending on your view, click on **Network and Internet.**



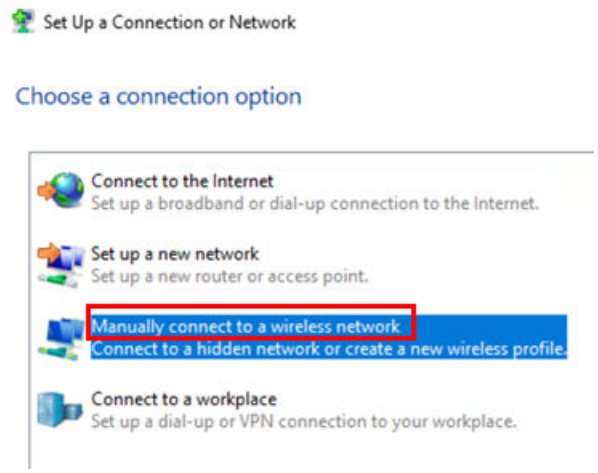
Depending on your view, click on **Network and Sharing Center.**



Click on **Set up a new connection or network.**

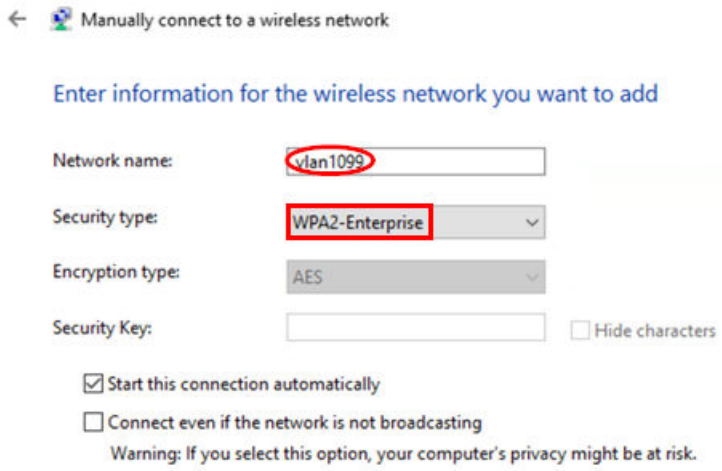


Select **Manually connect to a wireless network**.

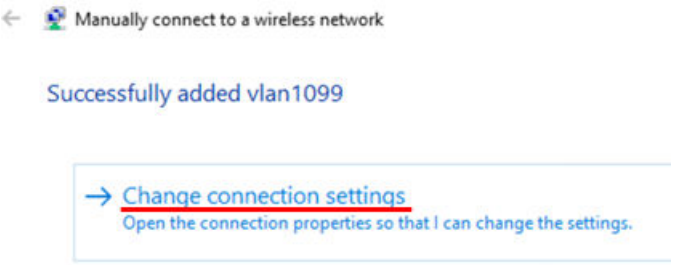


Configure the following settings:

- Network Name=`<SSID>`
- Security type=`WPA2-Enterprise`



Select **Change connection settings.**

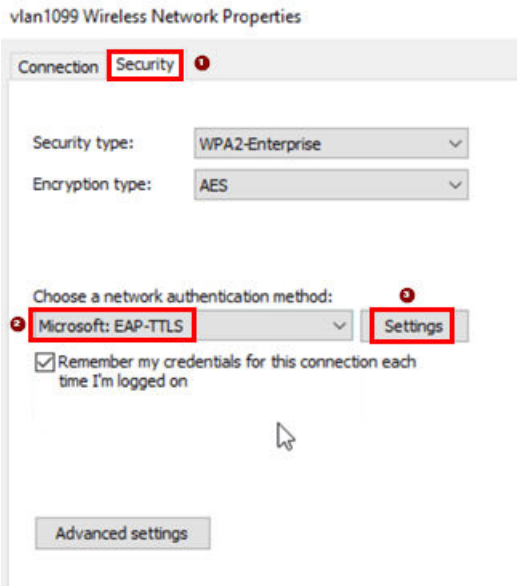


Change to the **Security** tab. Then, configure the following settings:

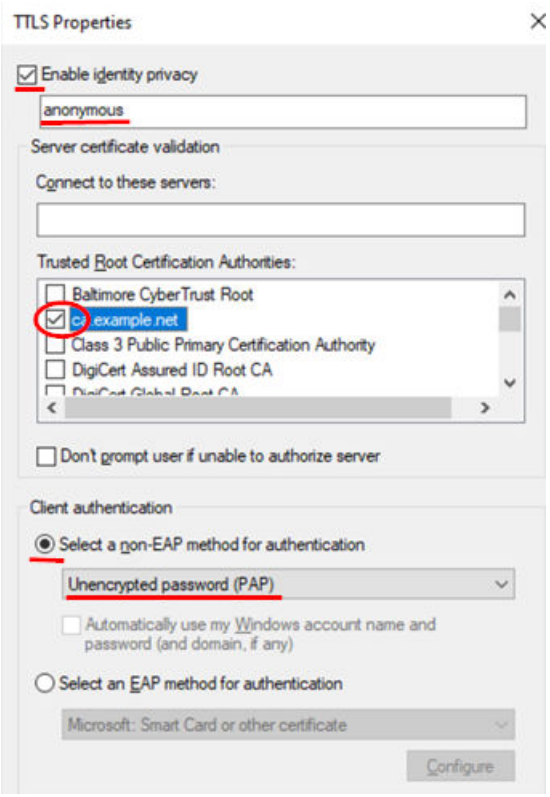
- Network authentication method=

```
Microsoft: Smart Card or
other certificate
```

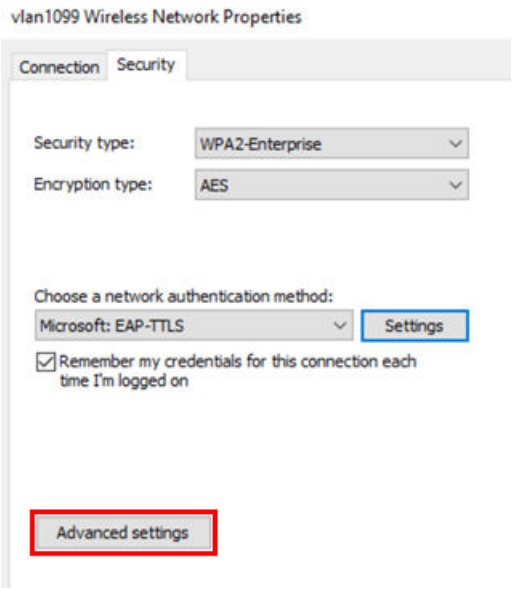
- Click on=`Settings`



The only change in this dialogue is to select our **Trusted Root Certification Authorities,** which in our case is `ca.example.net`. Leave the remaining fields as their defaults and return to the previous dialogue. Make sure the client authentication method is the default **Unencrypted password (PAP).**



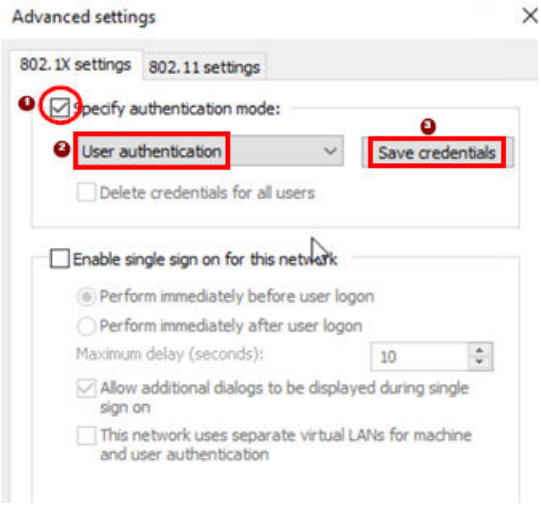
Click on **Additional Settings**.



Configure the following settings.

- Specify authentication mode=`Checked`
- Mode=`User authentication`

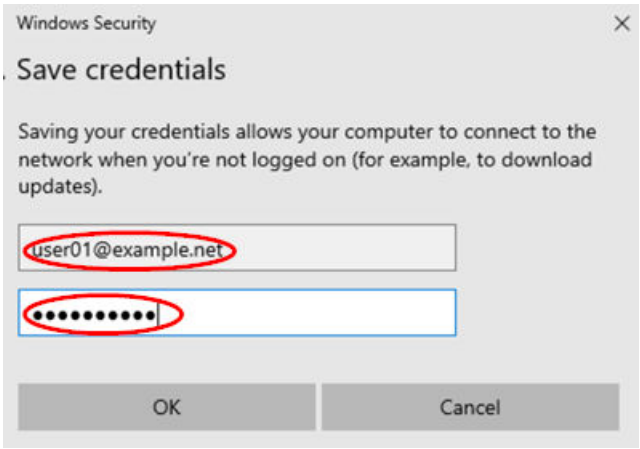
- Click-on=`Save credentials`



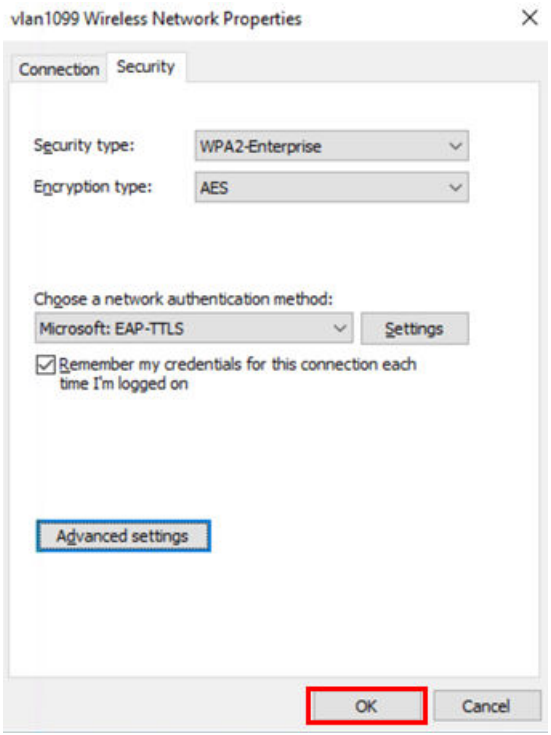
Next, enter a username and password that is valid for the remote IdP the Juniper Mist authentication cloud will contact for validation. In our case, with the example PKI, enter the following:

- Username=`user01@example.net`
- Password=`juniper123`

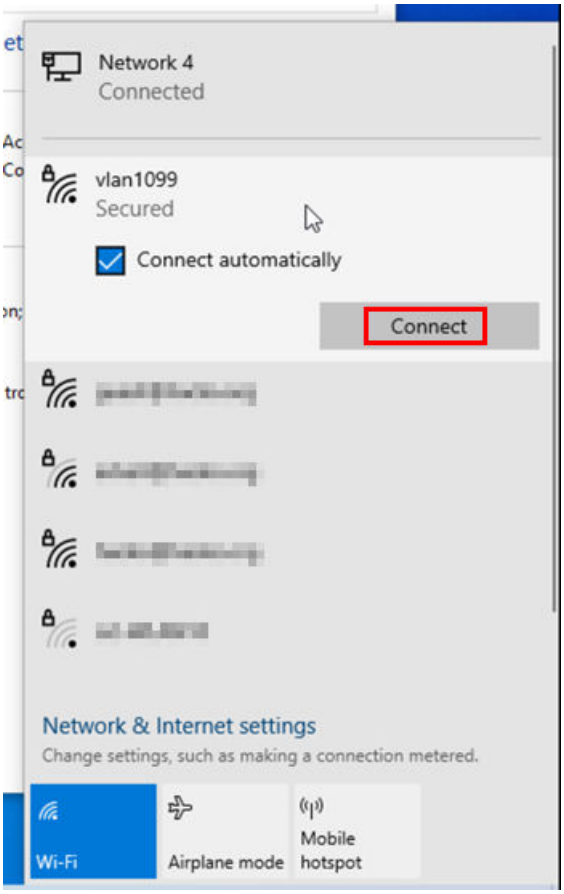
Then, save this dialogue and return to the previous one.



You can now finish the main dialogue.



Next, connect to the wireless network using the usual dialogue.



Linux Desktop Client

Linux does not support a dedicated certificate storage. We just transfer the files onto the system and make them known in the `/etc/wpa_supplicant/wpa_supplicant.conf` file which also contains the EAP authentication method configuration. In the examples in this chapter, we always perform the following when testing:

- Login to a desktop client VM.
- Load the required certificates from external storage.
- Write a new file with our configuration into `/etc/wpa_supplicant/wpa_supplicant.conf`
- Start the EAP supplicant in the foreground to see any debugging messages.

> **NOTE**: When using EAP-TLS, the user certificate also contains the entire path to the root CA, hence no extra configuration is needed.

Linux Client Wired EAP-TLS Example

In our example, `ens5` is the ethernet interface.

```
              virsh console desktop1
# load your pkcs user-file from Lab-Host into VM
scp root@192.168.10.1:examplePKI/user01.p12 .
# configure the WPA-Supplicant for EAP-TLS
cat <<EOF >/etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant
ctrl_interface_group=wheel
eapol_version=2
ap_scan=0
network={
   key_mgmt=IEEE8021X
   eap=TLS
   identity="user01@example.net"
   private_key="/root/user01.p12"
   private_key_passwd="juniper123"
   eapol_flags=0
}
EOF
# now start the wpa-supplicant in forground to see its messages
wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -D wired -i ens5
Successfully initialized wpa_supplicant
ens5: Associated with 01:80:c2:00:00:03
ens5: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
ens5: CTRL-EVENT-EAP-STARTED EAP authentication started
ens5: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=13
ens5: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 13 (TLS) selected
ens5: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/O=Juniper/OU=CA-
Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
ens5: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/O=Juniper/OU=CA-
Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
ens5: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/C=US/ST=California/O=Example TEST-Corp./OU=IT-
Department/CN=radius.example.net'
hash=3fa3a749a49e49597c3fd9b2441f0c4ce6ab4e9e792868ac94a82fe69f0768c9
ens5: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:radius.example.net
ens5: CTRL-EVENT-EAP-SUCCESS EAP authentication completed successfully
ens5: CTRL-EVENT-CONNECTED - Connection to 01:80:c2:00:00:03 completed [id=0 id_str=]
```

Linux Client Wireless EAP-TLS Example

```
              virsh console desktop2
# load your pkcs user-file from Lab-Host into VM
scp root@192.168.10.1:examplePKI/user01.p12 .
# configure the WPA-Supplicant for EAP-TLS
cat <<EOF >/etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant
ctrl_interface_group=wheel
eapol_version=2
ap_scan=1
network={
   ssid="vlan1099"               # SSID of the network to connect
   scan_ssid=1                   # enable probe request for finding networks using hidden SSID
   key_mgmt=WPA-EAP              # use external authentication, not pre-shared key
   eap=TLS
   identity="user01@example.net"
   private_key="/root/user01.p12"
   private_key_passwd="juniper123"
   eapol_flags=0
}
EOF
# check how your WLAN USB-Adapter is named
iwconfig
lo        no wireless extensions.
wlxe8de27a0e68e  IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry short limit:7   RTS thr=2347 B   Fragment thr:off
          Encryption key:off
          Power Management:off
ens3      no wireless extensions.
# now start the wpa-supplicant in forground to see its messages
rm -f /var/run/wpa_supplicant/wlxe8de27a0e68e
              wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -D nl80211 -i
wlxe8de27a0e68e
Successfully initialized wpa_supplicant
wlxe8de27a0e68e: SME: Trying to authenticate with d4:20:b0:11:56:13 (SSID='vlan1099' freq=2462
MHz)
wlxe8de27a0e68e: Trying to associate with d4:20:b0:11:56:13 (SSID='vlan1099' freq=2462 MHz)
wlxe8de27a0e68e: Associated with d4:20:b0:11:56:13
wlxe8de27a0e68e: CTRL-EVENT-EAP-STARTED EAP authentication started
```

```
wlxe8de27a0e68e: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlxe8de27a0e68e: CTRL-EVENT-REGDOM-CHANGE init=COUNTRY_IE type=COUNTRY alpha2=US
wlxe8de27a0e68e: CTRL-EVENT-EAP-STARTED EAP authentication started
wlxe8de27a0e68e: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=13
wlxe8de27a0e68e: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 13 (TLS) selected
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/
O=Juniper/OU=CA-Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/
O=Juniper/OU=CA-Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/C=US/ST=California/O=Example TEST-
Corp./OU=IT-Department/CN=radius.example.net'
hash=3fa3a749a49e49597c3fd9b2441f0c4ce6ab4e9e792868ac94a82fe69f0768c9
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:radius.example.net
wlxe8de27a0e68e: CTRL-EVENT-EAP-SUCCESS EAP authentication completed successfully
wlxe8de27a0e68e: PMKSA-CACHE-ADDED d4:20:b0:11:56:13 0
wlxe8de27a0e68e: WPA: Key negotiation completed with d4:20:b0:11:56:13 [PTK=CCMP GTK=CCMP]
wlxe8de27a0e68e: CTRL-EVENT-CONNECTED - Connection to d4:20:b0:11:56:13 completed [id=0 id_str=]
```

Linux Client Wired EAP-TTLS Example

In our example, `ens5` is the ethernet interface:

```
              virsh console desktop1
# load your root-ca-file from Lab-Host into VM
scp root@192.168.10.1:examplePKI/cacert.pem .
# configure the WPA-Supplicant for EAP-TTLS w. PAP
cat <<EOF >/etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant
ctrl_interface_group=wheel
ap_scan=0
network={
  key_mgmt= IEEE8021X           # use external authentication, not pre-shared key
  eap=TTLS                      # enable TTLS method for encrypted tunnel
  ca_cert="/root/cacert.pem"    # root-ca that signed the cert of the AAA-Server
  anonymous_identity="anon@example.net"    # use identity outside tunnel as NAI
  phase2="auth=PAP"             # use password authentication protocol
                                # this will pass clear-text password inside tunnel
  identity="user01@example.net" # use this identity inside tunnel
  password="juniper123"         # password of the authenticating user
```

```
}
EOF
# now start the wpa-supplicant in forground to see its messages
wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -D wired -i ens5
Successfully initialized wpa_supplicant
ens5: Associated with 01:80:c2:00:00:03
ens5: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
ens5: CTRL-EVENT-EAP-STARTED EAP authentication started
ens5: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=13 -> NAK
ens5: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21
ens5: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 21 (TTLS) selected
ens5: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/O=Juniper/OU=CA-
Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
ens5: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/C=US/ST=California/O=Example TEST-Corp./OU=IT-
Department/CN=radius.example.net'
hash=3fa3a749a49e49597c3fd9b2441f0c4ce6ab4e9e792868ac94a82fe69f0768c9
ens5: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:radius.example.net
ens5: CTRL-EVENT-EAP-SUCCESS EAP authentication completed successfully
ens5: CTRL-EVENT-CONNECTED - Connection to 01:80:c2:00:00:03 completed [id=0 id_str=]
```

Linux Client Wireless EAP-TTLS Example

```
              virsh console desktop2
# load your root-ca-file from Lab-Host into VM
scp root@192.168.10.1:examplePKI/cacert.pem .
# configure the WPA-Supplicant for EAP-TTLS w. PAP
cat <<EOF >/etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant
ctrl_interface_group=wheel
ap_scan=1
network={
  ssid="vlan1099"              # SSID of the network to connect
  scan_ssid=1                  # enable probe request for finding networks using hidden SSID
  key_mgmt=WPA-EAP             # use external authentication, not pre-shared key
  eap=TTLS                     # enable TTLS method for encrypted tunnel
  ca_cert="/root/cacert.pem"   # root-ca that signed the cert of the AAA-Server
  anonymous_identity="anon@example.net"    # use identity outside tunnel as NAI
  phase2="auth=PAP"            # use password authentication protocol
                               # this will pass clear-text password inside tunnel
  identity="user01@example.net" # use this identity inside tunnel
```

```
  password="juniper123"         # password of the authenticating user
}
EOF
# check how your WLAN USB-Adapter is named
iwconfig
lo        no wireless extensions.
wlxe8de27a0e68e  IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry short limit:7   RTS thr=2347 B   Fragment thr:off
          Encryption key:off
          Power Management:off
ens3      no wireless extensions.
# now start the wpa-supplicant in forground to see its messages
rm -f /var/run/wpa_supplicant/wlxe8de27a0e68e
              wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -D nl80211 -i
wlxe8de27a0e68e
Successfully initialized wpa_supplicant
wlxe8de27a0e68e: SME: Trying to authenticate with d4:20:b0:11:56:13 (SSID='vlan1099' freq=2437
MHz)
wlxe8de27a0e68e: Trying to associate with d4:20:b0:11:56:13 (SSID='vlan1099' freq=2437 MHz)
wlxe8de27a0e68e: Associated with d4:20:b0:11:56:13
wlxe8de27a0e68e: CTRL-EVENT-EAP-STARTED EAP authentication started
wlxe8de27a0e68e: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlxe8de27a0e68e: CTRL-EVENT-REGDOM-CHANGE init=COUNTRY_IE type=COUNTRY alpha2=US
wlxe8de27a0e68e: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=13 -> NAK
wlxe8de27a0e68e: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21
wlxe8de27a0e68e: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 21 (TTLS) selected
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-CERT depth=1 subject='/C=NL/ST=Netherlands/L=Amsterdam/
O=Juniper/OU=CA-Center/CN=ca.example.net/emailAddress=trustcenter@example.net'
hash=6474bda8fe419b2525f6efa3579d2947437bc08e1a8ded9d48724f610c7c50e5
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/C=US/ST=California/O=Example TEST-
Corp./OU=IT-Department/CN=radius.example.net'
hash=3fa3a749a49e49597c3fd9b2441f0c4ce6ab4e9e792868ac94a82fe69f0768c9
wlxe8de27a0e68e: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:radius.example.net
wlxe8de27a0e68e: CTRL-EVENT-EAP-SUCCESS EAP authentication completed successfully
wlxe8de27a0e68e: PMKSA-CACHE-ADDED d4:20:b0:11:56:13 0
wlxe8de27a0e68e: WPA: Key negotiation completed with d4:20:b0:11:56:13 [PTK=CCMP GTK=CCMP]
wlxe8de27a0e68e: CTRL-EVENT-CONNECTED - Connection to d4:20:b0:11:56:13 completed [id=0 id_str=]
```

Configuration Examples of Public Identity Provider Database Integration

Azure AD Integration

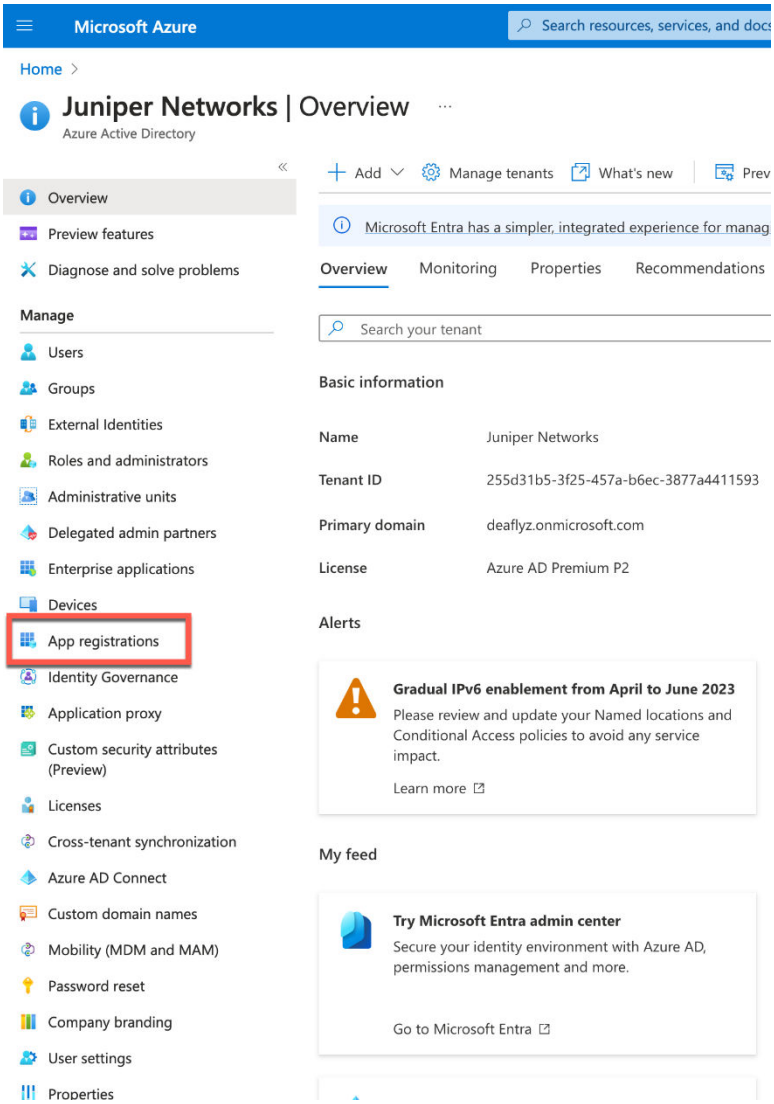
Juniper Mist Access Assurance allows you to integrate our authentication service natively into Azure Active Directory using OAuth. Then, you leverage Azure AD as an IdP in combination with Juniper Mist Access Assurance in the following way:

- User authentication with EAP-TTLS
  - Authenticate users by doing delegated authentication checking username and password via OAuth
  - Obtain user group memberships to leverage them in auth policies
  - Obtain user account state information (active or suspended)
- User authorization with EAP-TLS or EAP-TTLS
  - Obtain user account state information (active or suspended)
  - Obtain user group memberships to leverage them in auth policies
- User authorization status by MDM compliance status (using Microsoft Intune)

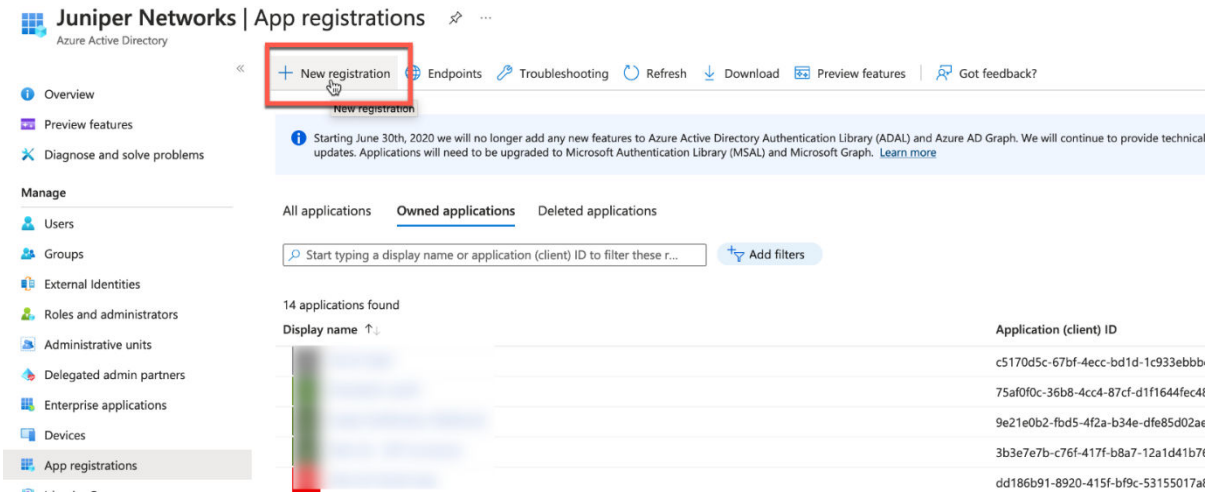
Azure AD Integration Part 1: Azure Portal configuration

Login to your Azure tenant at `portal.azure.com`.

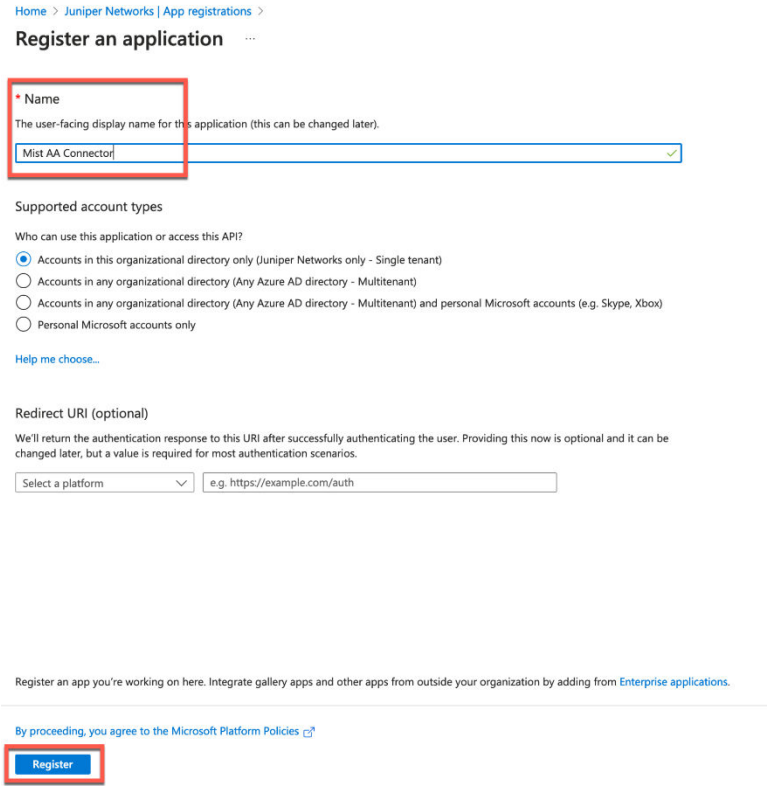
Navigate to **Azure Active Directory > App Registrations**.



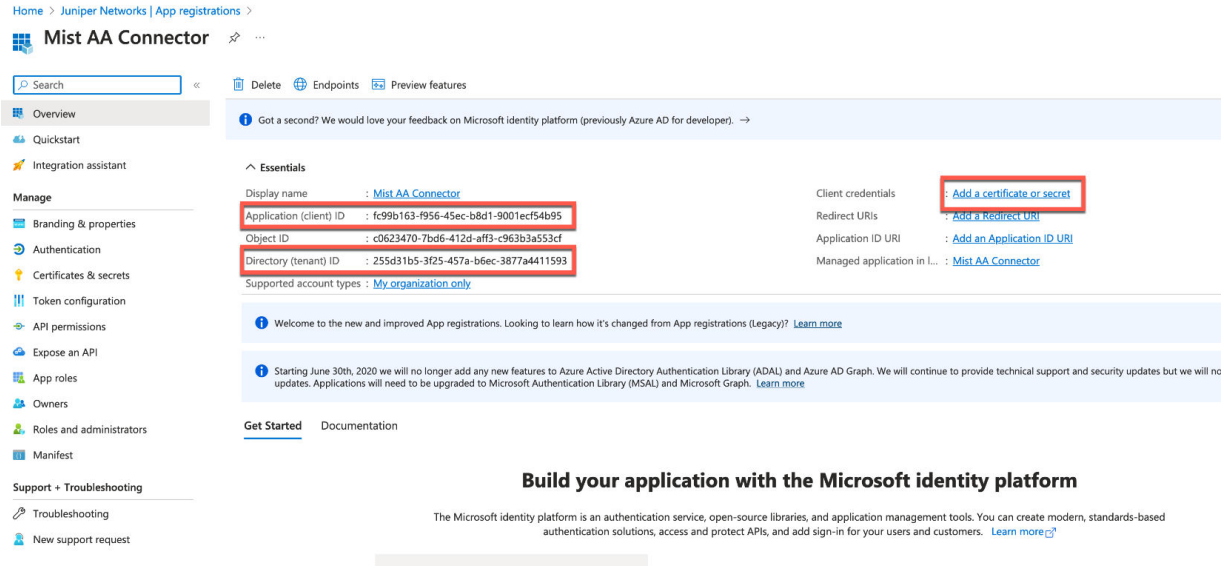
Click on **New Registration**.



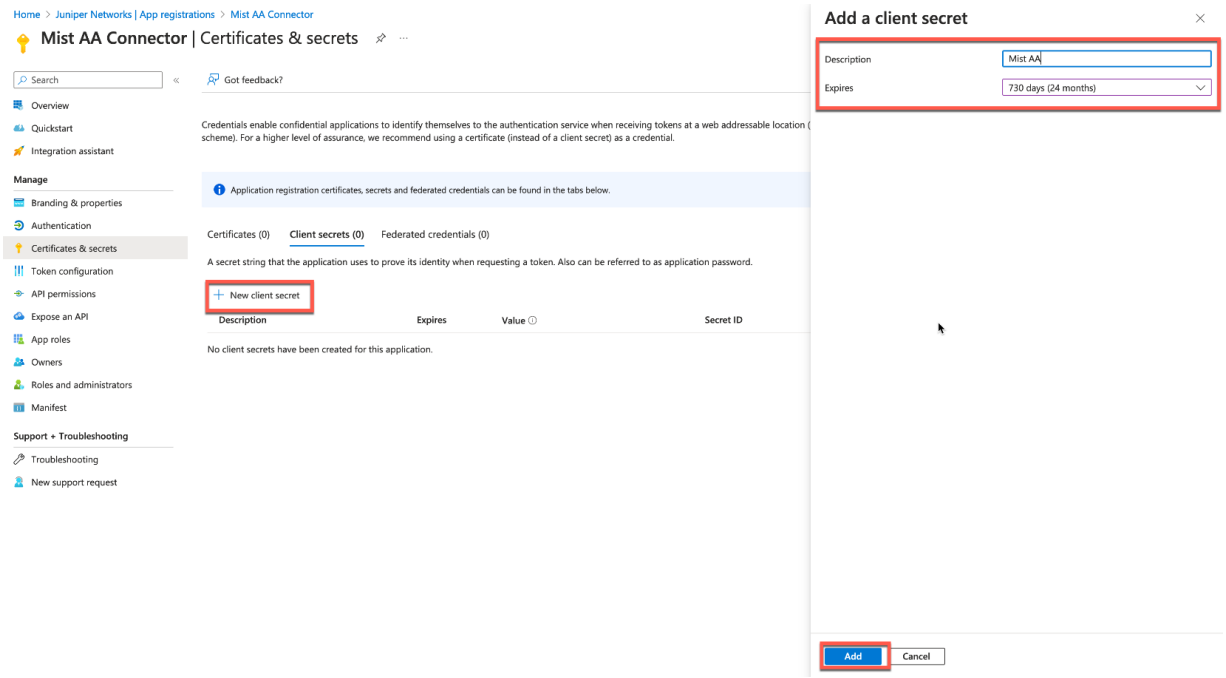
Give your application a name, then click **Register**.



Copy and save **Application (client) ID, Directory (tenant) ID**, then select **Add a certificate or secret**.

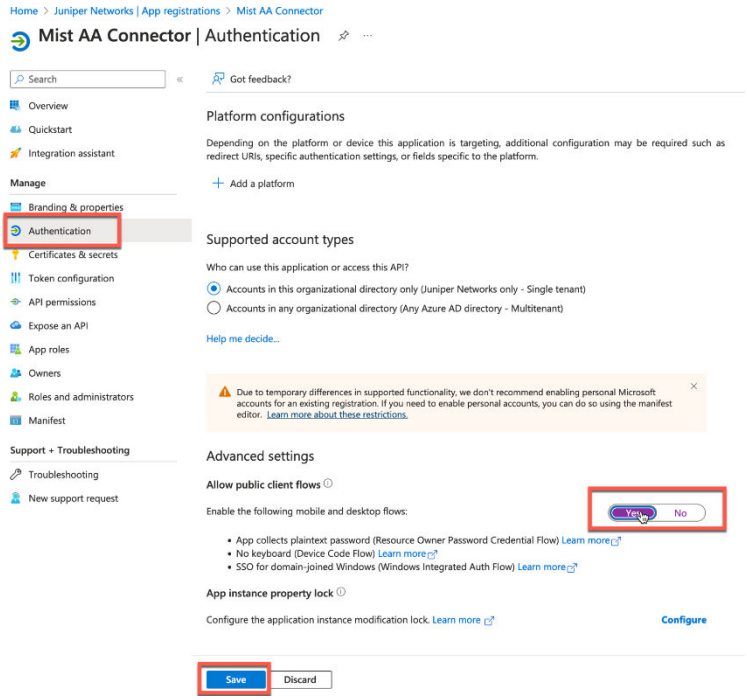


Select **New Client Secret**, set the secret expiration, then click **Add**.



Once the secret is added, copy and save the **Value** field (note that you will only see it once immediately after the secret is created, so make sure to save it in a secure place).

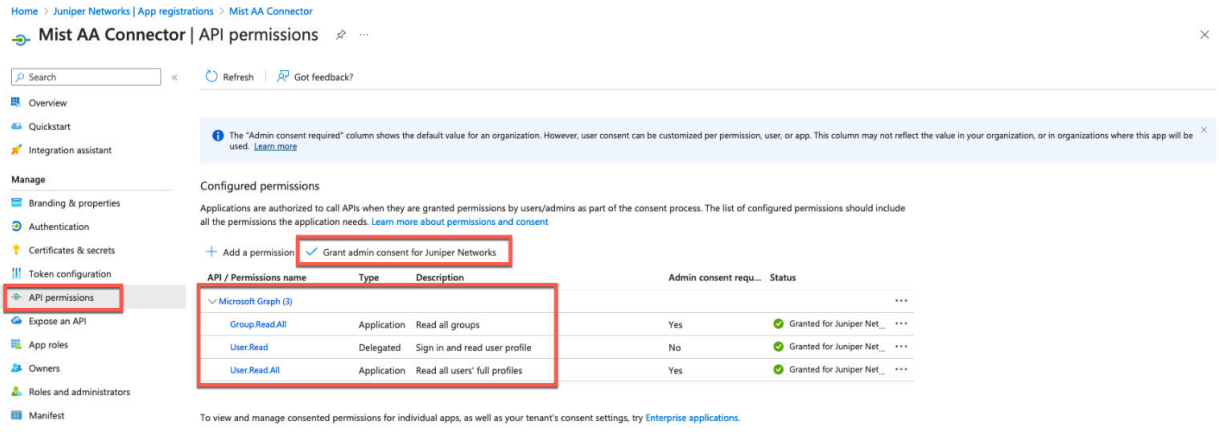
Now, navigate to the **Authentication** tab and enable **Allow public client flows**, this is required if you want to support EAP-TTLS credential-based authentication.



Lastly, navigate to the **API permissions** tab and grant the following permissions, and do not forget to grant admin consent:

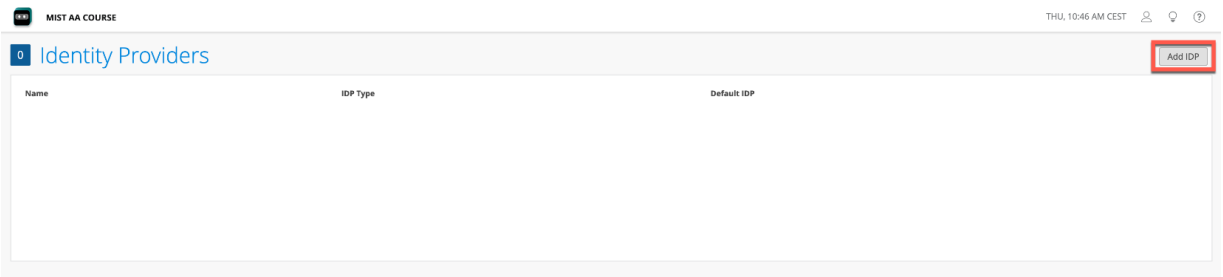
Microsoft Graph:

- User.Read=`Delegated`
- User.Read.All=`Application`
- Group.Read.All=`Application`



Azure AD Integration Part 2: Juniper Mist Cloud Configuration

Navigate to **Organization > Access > Identity Providers** and click **Add IDP**.



Configure Azure AD integration as follows:

- IDP Type=`OAuth`
- OAuth Type=`Azure`
- OAuth Tenant ID=

```
<paste value from Directory (tenant)
ID you copied from Azure app>
```

- Domain Names=

```
<configure your Azure domain
name(s)>
```

- OAuth Client Credential (CC) Client Id:=

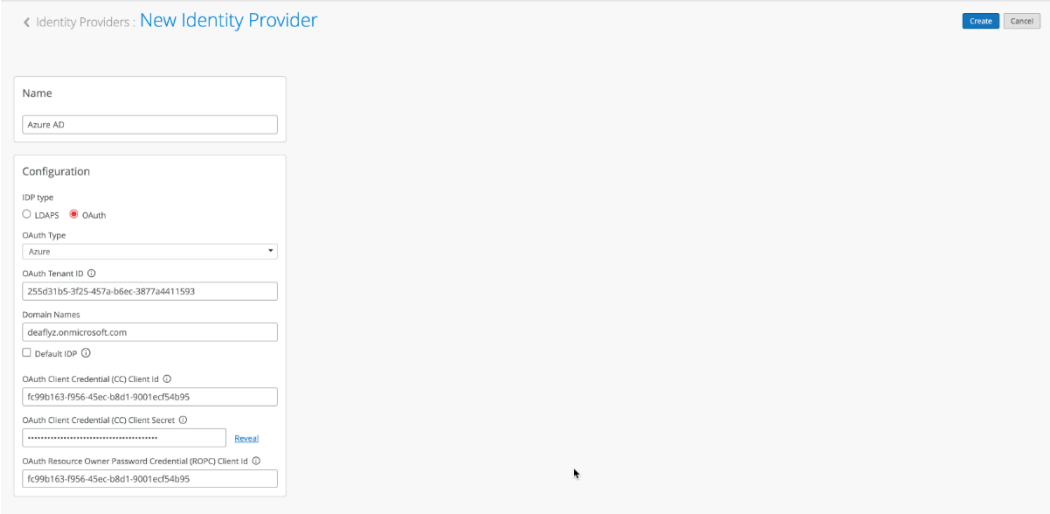
```
<paste
Application (client) ID you copied from Azure app
earlier>
```

- OAuth Client Credential (CC) Client Secret=

```
<paste
Value of the secret your created earlier>
```

- OAuth ROPC Client Id=

```
<paste Application (client) ID
from Azure>
```



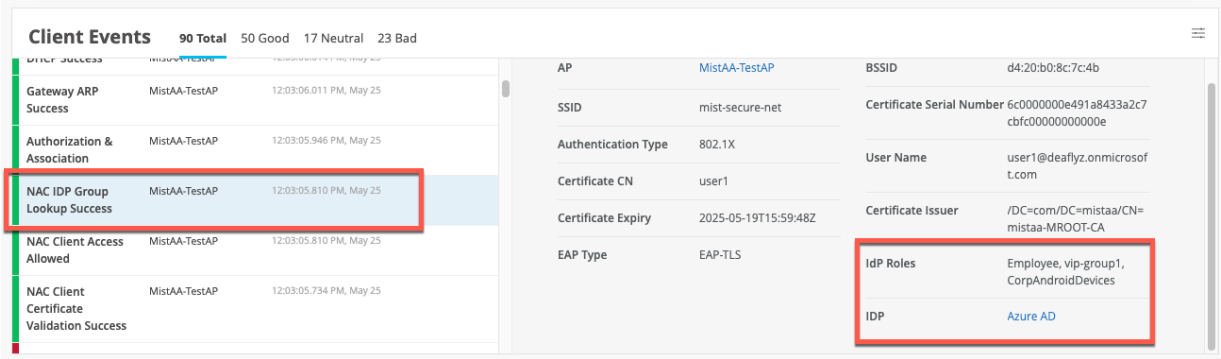
NOTE: Notes on EAP-TTLS authentication with Azure.

EAP-TTLS authentication leverages Resource Owner Password Credential (ROPC) OAuth flow with Azure AD, which means using legacy authentication using a username and password without multi-factor authentication (MFA). Below are a few considerations to keep in mind regarding using this method:

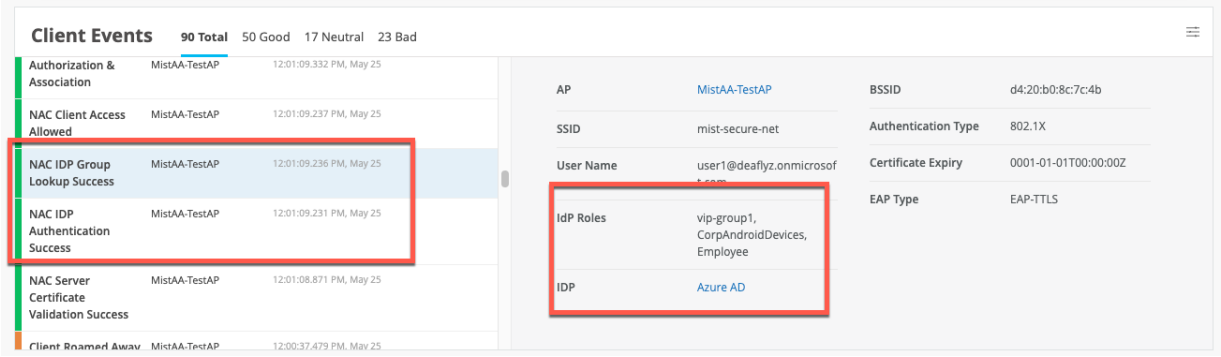
- Client devices need to be configured with the right Wi-Fi profile, either from group policy (GPO) or MDM—simply providing a username and password at the login prompt will not work for some operating systems.
- Users need to use their full user principal name (UPN) (username@domain), not just their username.
- Clients need to trust the server certificate. See following document for more details.
- Users need to log in at least once to Azure before ROPC authentication will work (this is important if using test user accounts).
- MFA needs to be disabled for users using this specific application, as MFA is not a real option for 802.1X (breaks roaming, and client timeouts).

Validation

When clients are authenticated using EAP-TLS with Azure AD lookup, you should see an additional event called **NAC IDP Group Lookup Success**.



In EAP-TTLS scenarios, you should see **NAC IDP Authentication Success** (indicating that Azure validated the user credentials), followed by a **NAC IDP Group Lookup Success** event that fetches user group memberships.



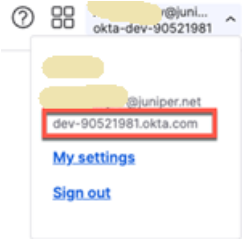
Okta Integration

Juniper Mist Access Assurance allows you to integrate our authentication service natively with the Okta directory using OAuth. Then, you leverage Okta as the IdP in combination with Juniper Mist Access Assurance in the following way:

- User authentication with EAP-TTLS
  - Authenticate the user by doing delegated authentication checking username and password via OAuth
  - Obtain user group memberships to leverage them in auth policies
  - Obtain user account state information (active or suspended)
- User authorization with EAP-TLS or EAP-TTLS
  - Obtain user account state information (active or suspended)
  - Obtain user group memberships to leverage them in auth policies

Okta Integration Part 1: Okta Dashboard

First, get your Okta tenant ID and save it. You can see it if you click in the upper-right corner of the Okta dashboard:

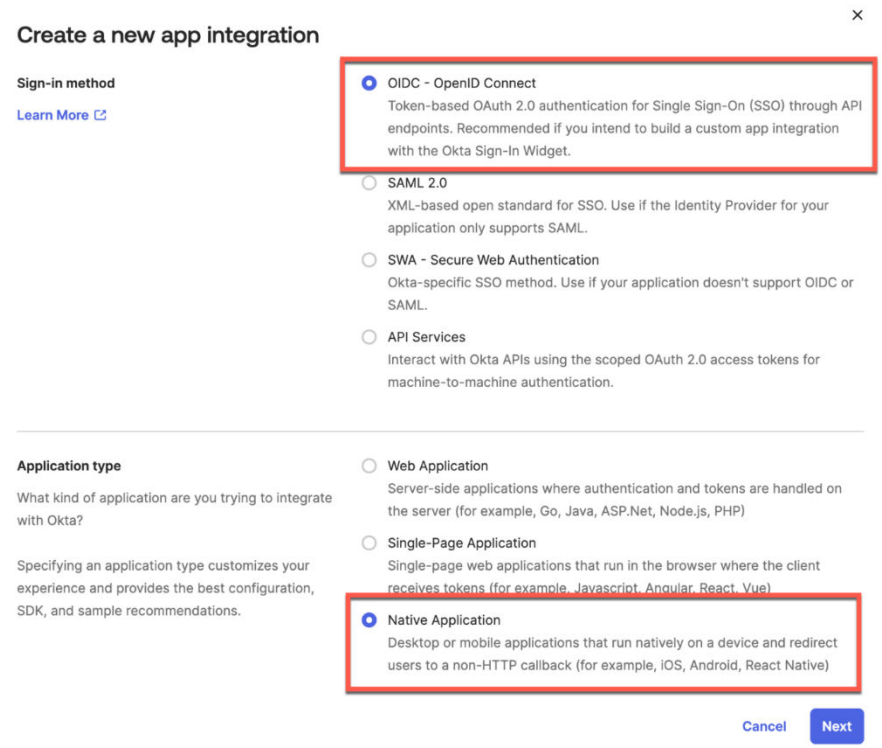


Okta Integration Part 2: OKTA Resource Owner Password Credential App Integration

> **NOTE**: The steps in this section are for the IdP to be able to validate the User credentials when asked by the Mist Authentication Cloud.

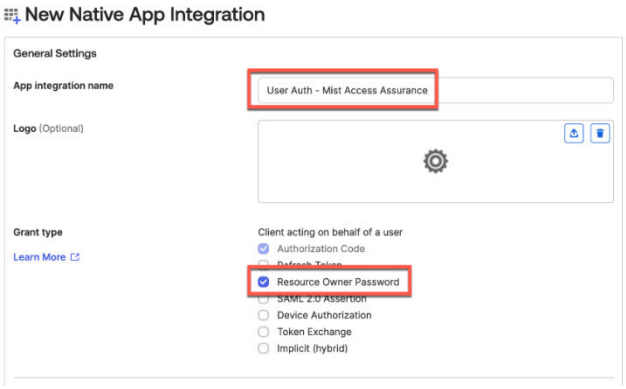
Step 1—Create App Integration

Navigate to **Applications > Create New App Integration**.

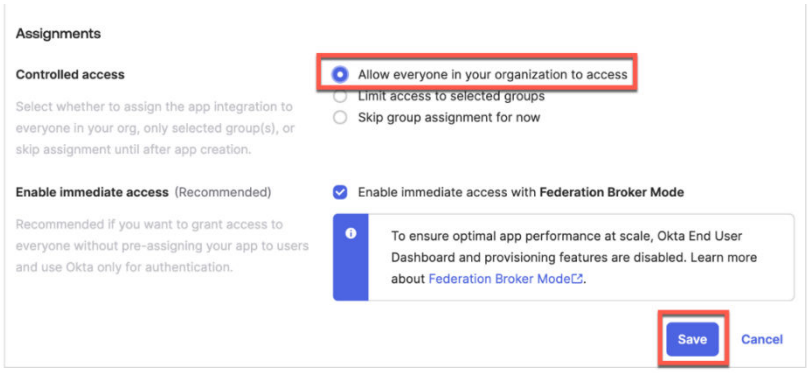


Step 2—Configure App Grant type and user assignments

Name your application, select **Resource Owner Password** grant type:

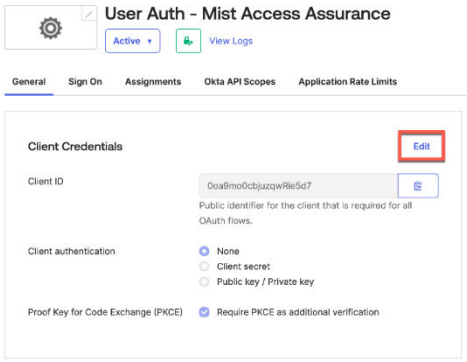


Select which users to allow to authenticate with this app.

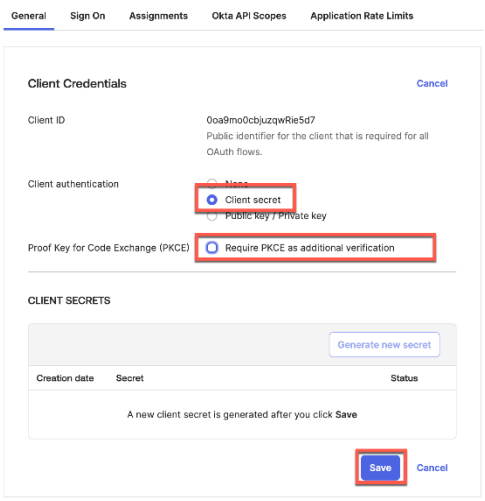


Step 3—Generate Client ID and Client Secret

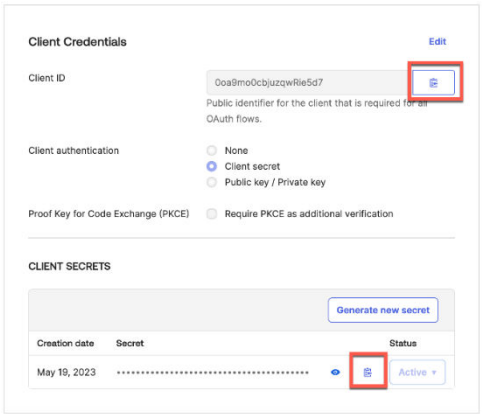
Let's generate the client secret. Under the **General** tab, click on the **Edit** button.



Set **Client Authentication** as **Client Secret**.

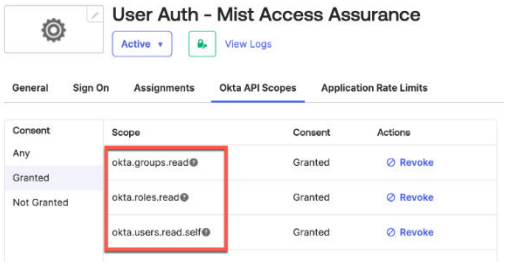


Copy the generated client ID and client secret.



Step 4—Configure App API Scopes

Go to the Okta API scopes to allow the app the following read permissions.

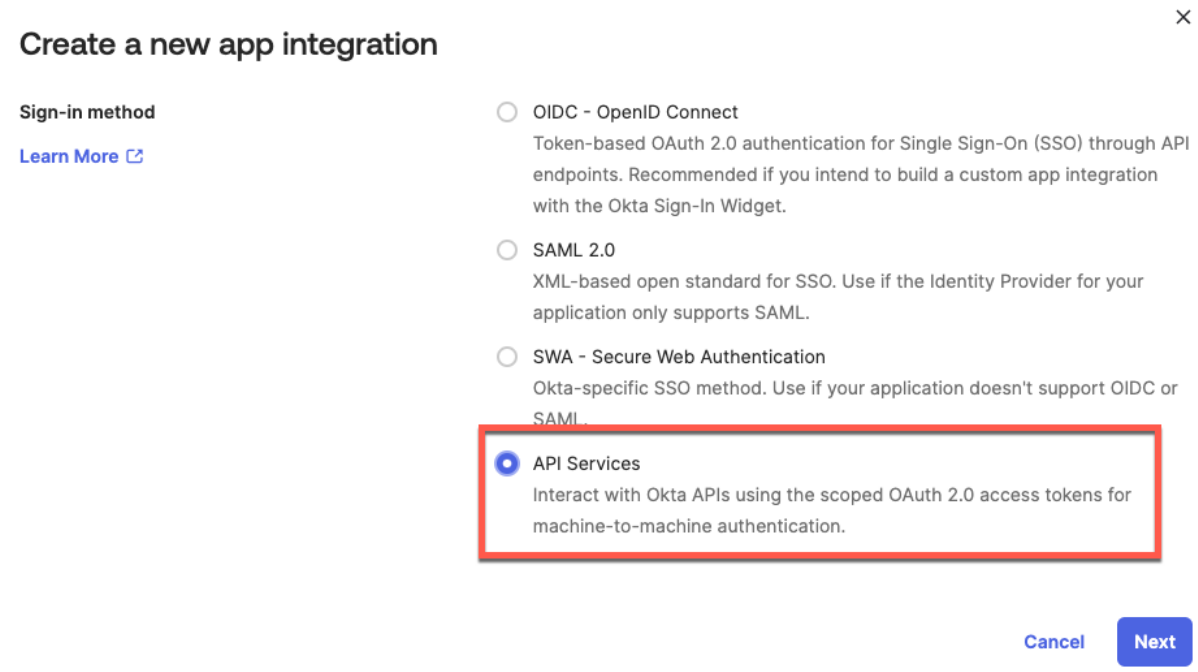


Okta Integration Part 3: OKTA Client Credential App Integration

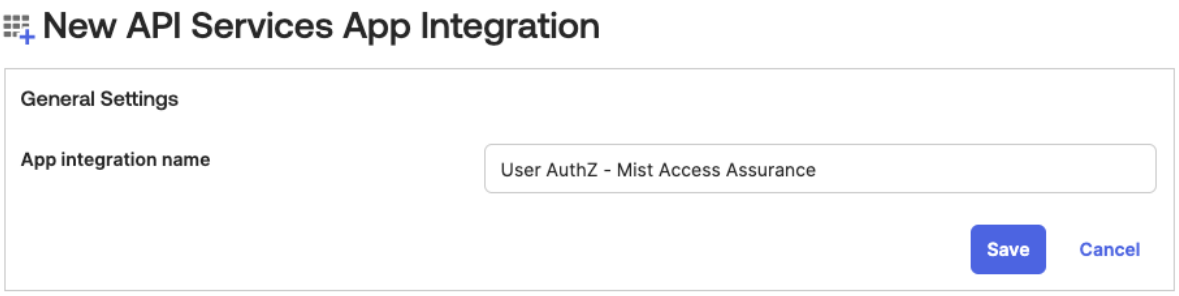
NOTE: The steps in this section are for the IdP to send back group assignments to the Mist Authentication cloud after successful user authentication.

Step 1—Create the App Integration

Navigate to **Applications > Create New App Integration**. Select API Services as the sign-in method.

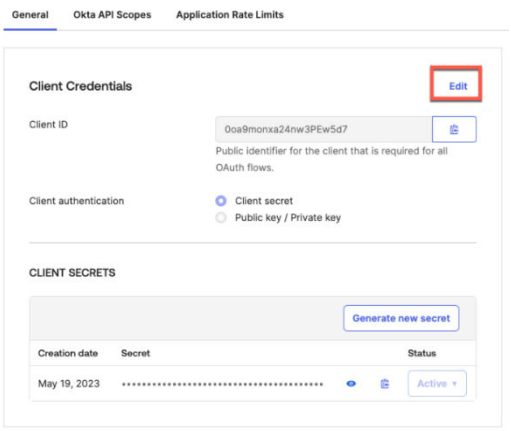


Name your application and click **Save**.

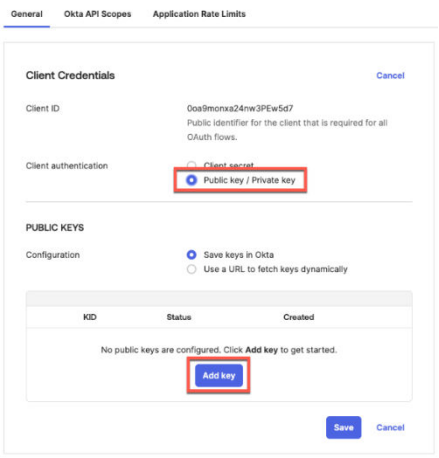


Step 2—Generate the Client ID and Private Key

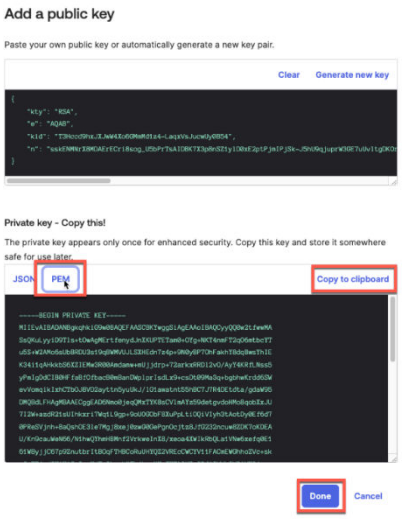
On the next screen, select **Edit** to generate a private key.



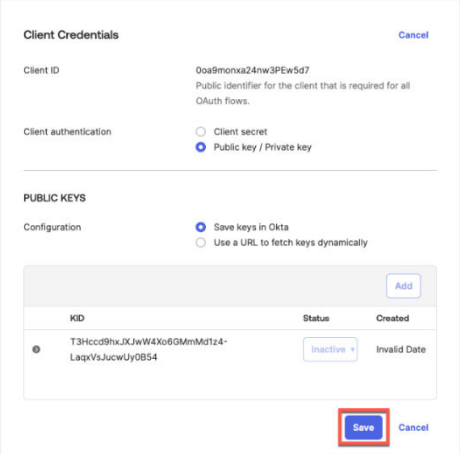
Set **Client Authentication** as **Public key / Private key**, then click on **Add key**.



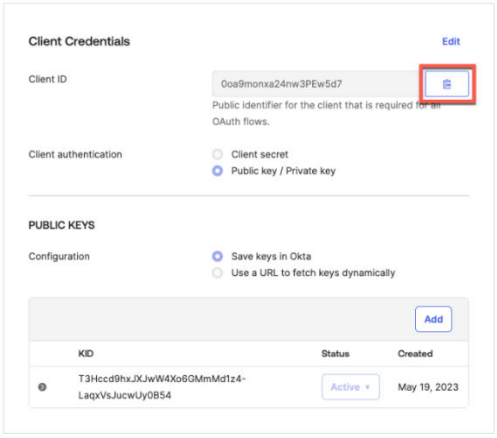
First, select **PEM** as the format, then click **Copy to clipboard** to copy the resulting private key (save it in a secure place), then click **Done**.



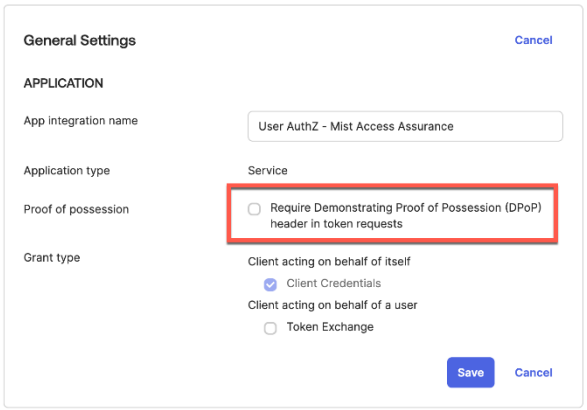
On the next screen, click **Save**.



Do not forget to copy and save your client ID.

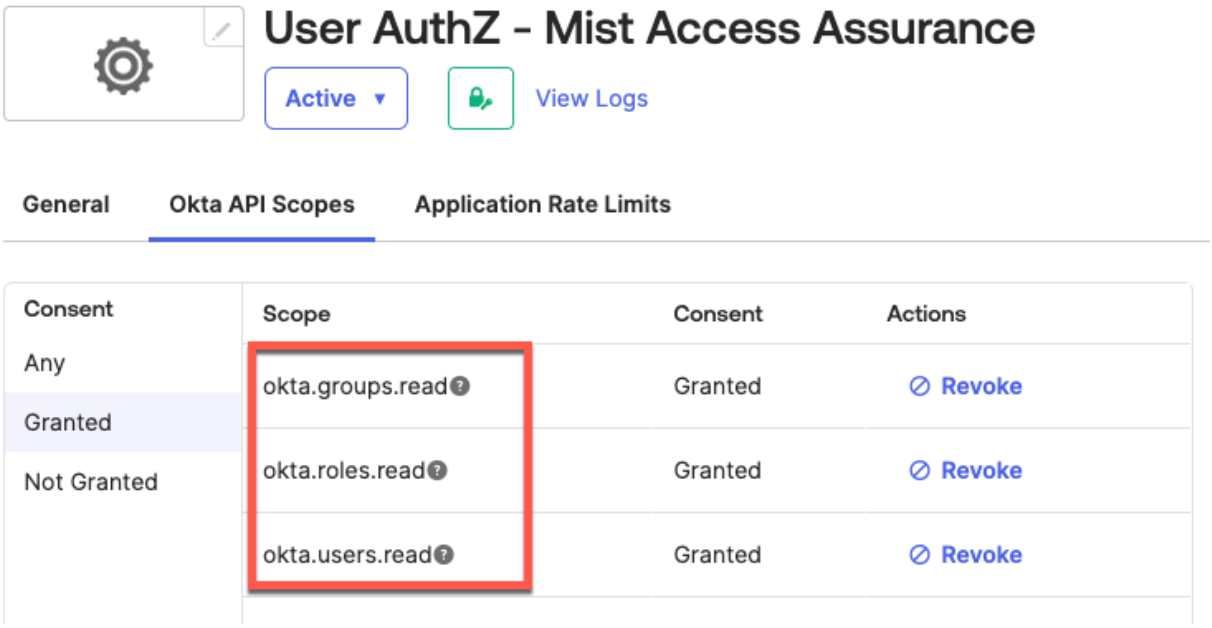


Scroll down until you see **General Settings** section, click Edit and uncheck “Require Demonstrating Proof of Possession (DPoP) header in token requests.”



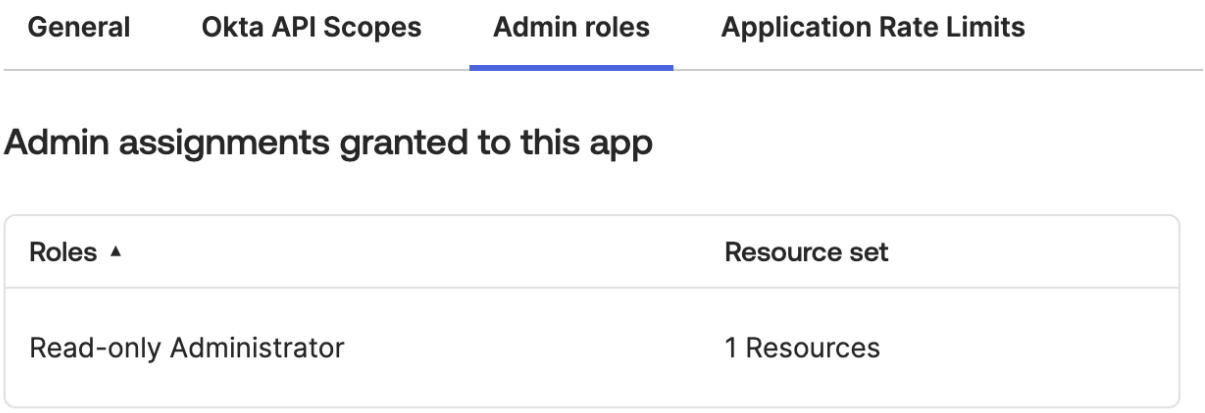
Step 3—Configure App API Scopes

Go to the **Okta API Scopes** tab and grant the following API permissions.



Step 4 – Assign Admin Role to App

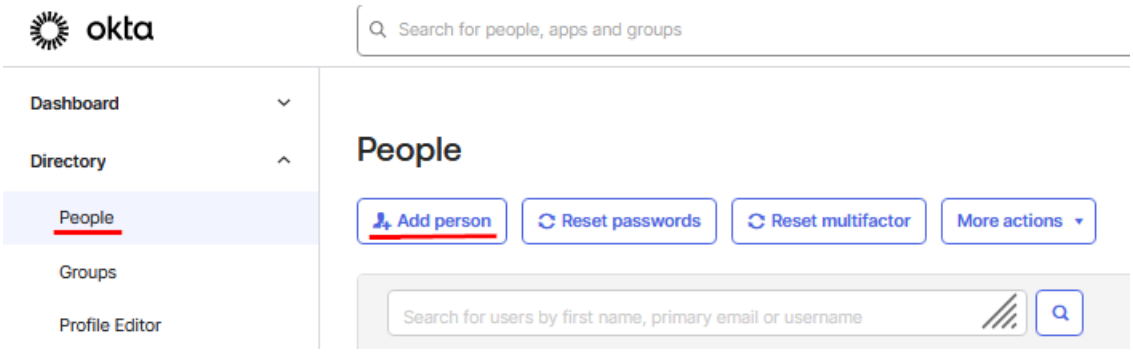
Navigate to **Admin roles** tab and assign Read-only Administrator role, otherwise app will not be able to read any data via API.



Okta Integration Part 4: Local Okta User Creation Example

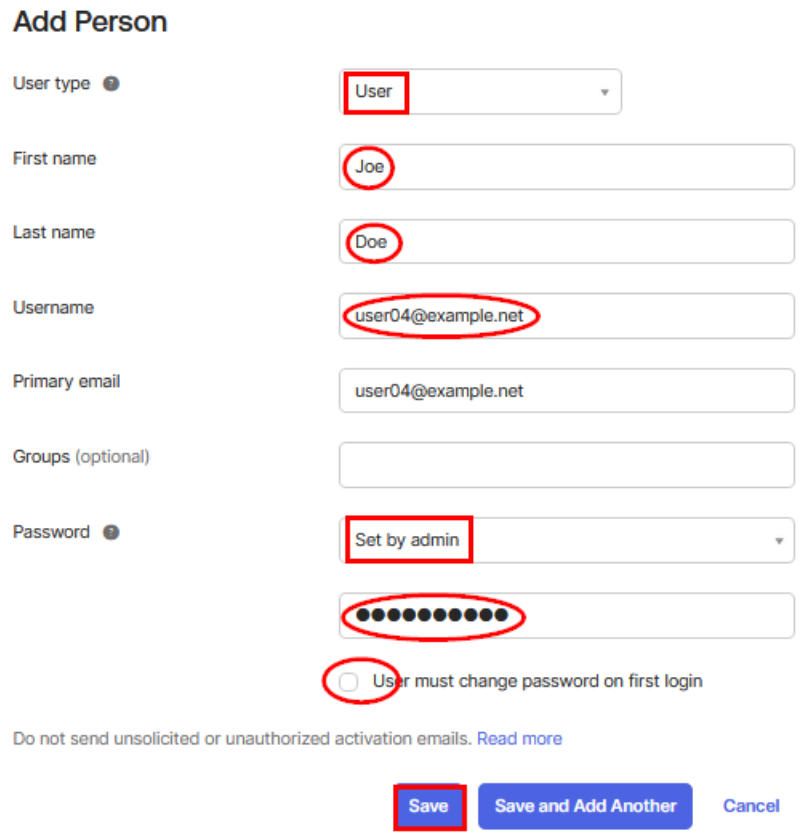
> **NOTE**: Local Okta Users creation is optional! You may use other methods to manage your User Accounts.

To add a new User Account got to **Directory > People > Add person**.

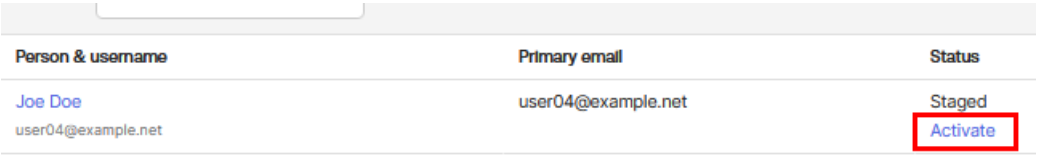


Configure the following example user, then save the account:

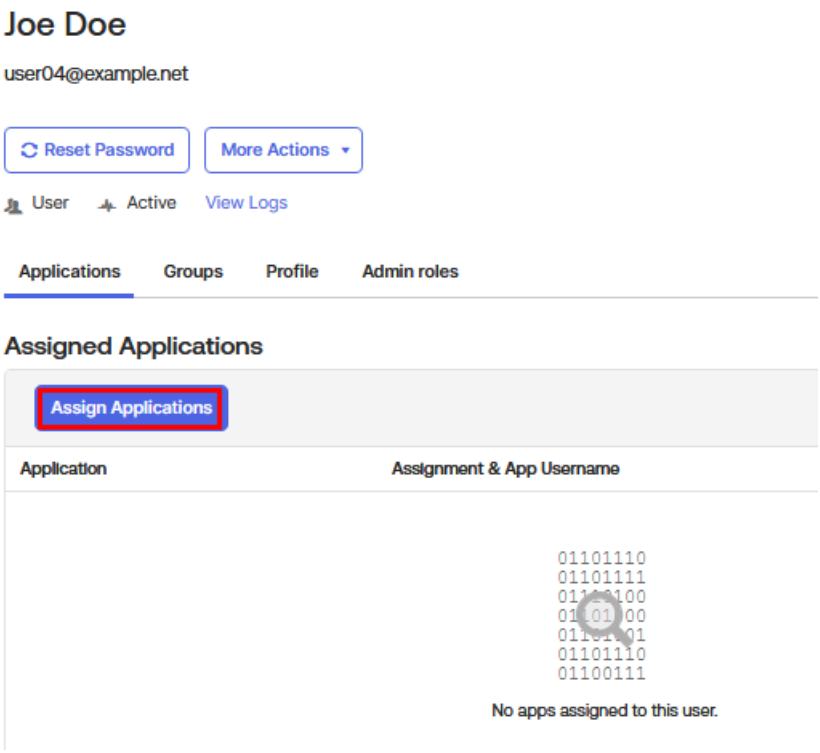
- User type=`User`
- First name=`Joe`
- Last name=`Doe`
- Username=user04@example.net (used for IdP login)
- Password
  - Password=`<your password>`
  - User must change password on first login=`Unchecked/Disabled`



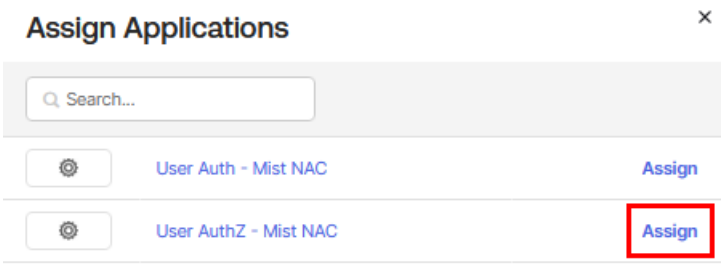
Then **Activate** the User Account



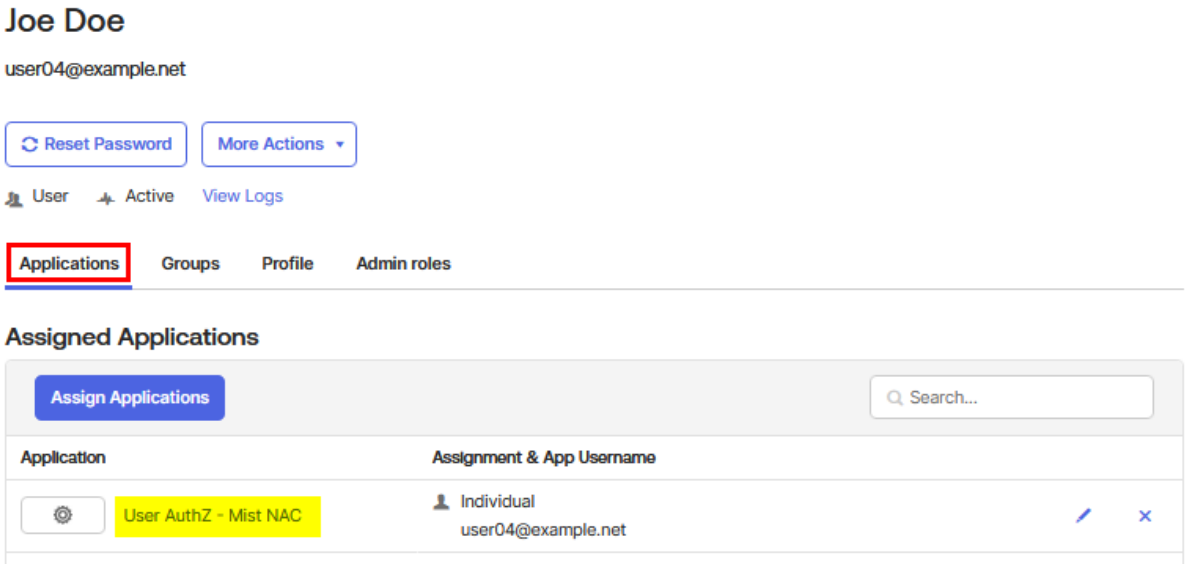
Assign the second Application to this User Account (above we use the name “User AuthZ – Mist Access Assurance”) to allow the readout of the group assigns for this account as part of the IdP.



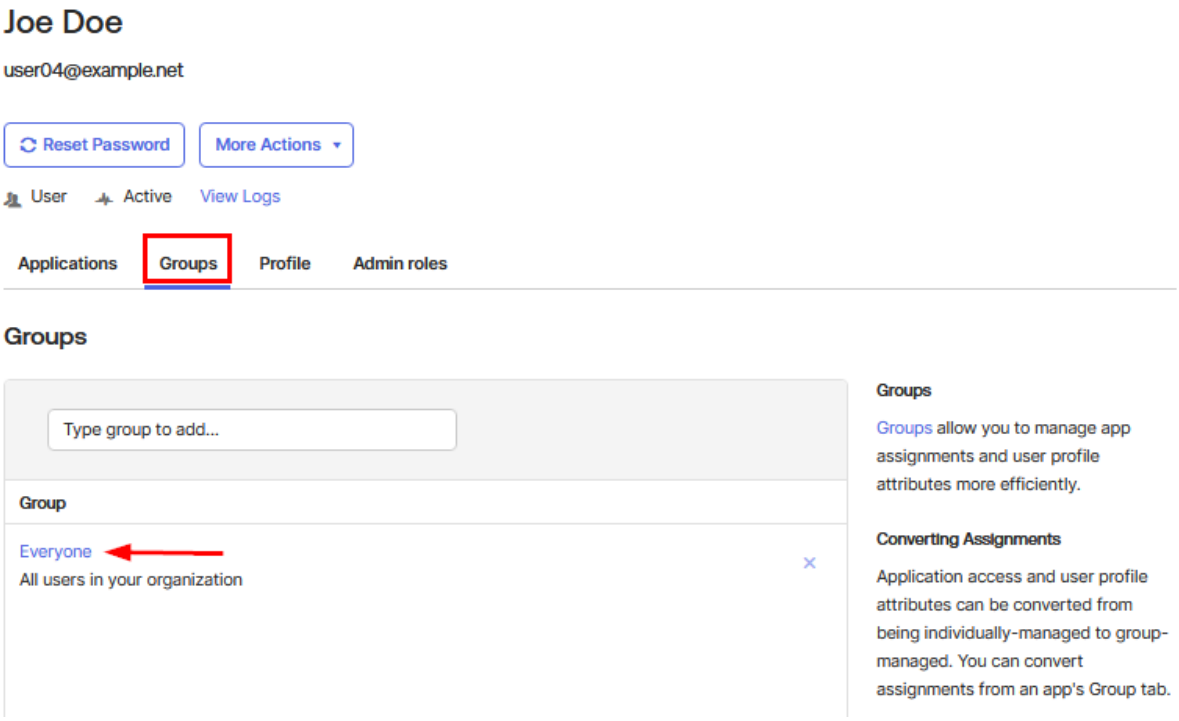
Assign the second Application.



Check the Application is now assigned to the User Account.



Check the Groups this User Account belongs to. The group `Everyone` is default. You can add more as required. All these groups will then be reported back to the Mist Authentication Cloud and one can make Label based decisions.



Okta Integration Part 5: Juniper Mist Cloud Configuration

First, get your Okta tenant ID into the new IdP under the Juniper Mist dashboard.

Step 1—Add Identity Provider

In the Juniper Mist dashboard, navigate to **Organization > Access > Identity Providers > Add**.

Configure the IdP as follows:

- IDP Type=`OAuth`
- OAuth Type=`Okta`
- OAuth Tenant ID=

```
<Okta Tenant ID (from Part
1)>
```

NOTE: Should you use an Okta developer account for testing then the OAuth Tenant ID must not contain a domain extension and look like “dev-<number>.”

- Domain Names=

```
<Your Okta users domain name(s), e.g.
company.com>
```

- OAuth CC Client ID=

```
<Client ID you copied in Part 3 /
Step 2>
```

- OAuth CC Private Key=

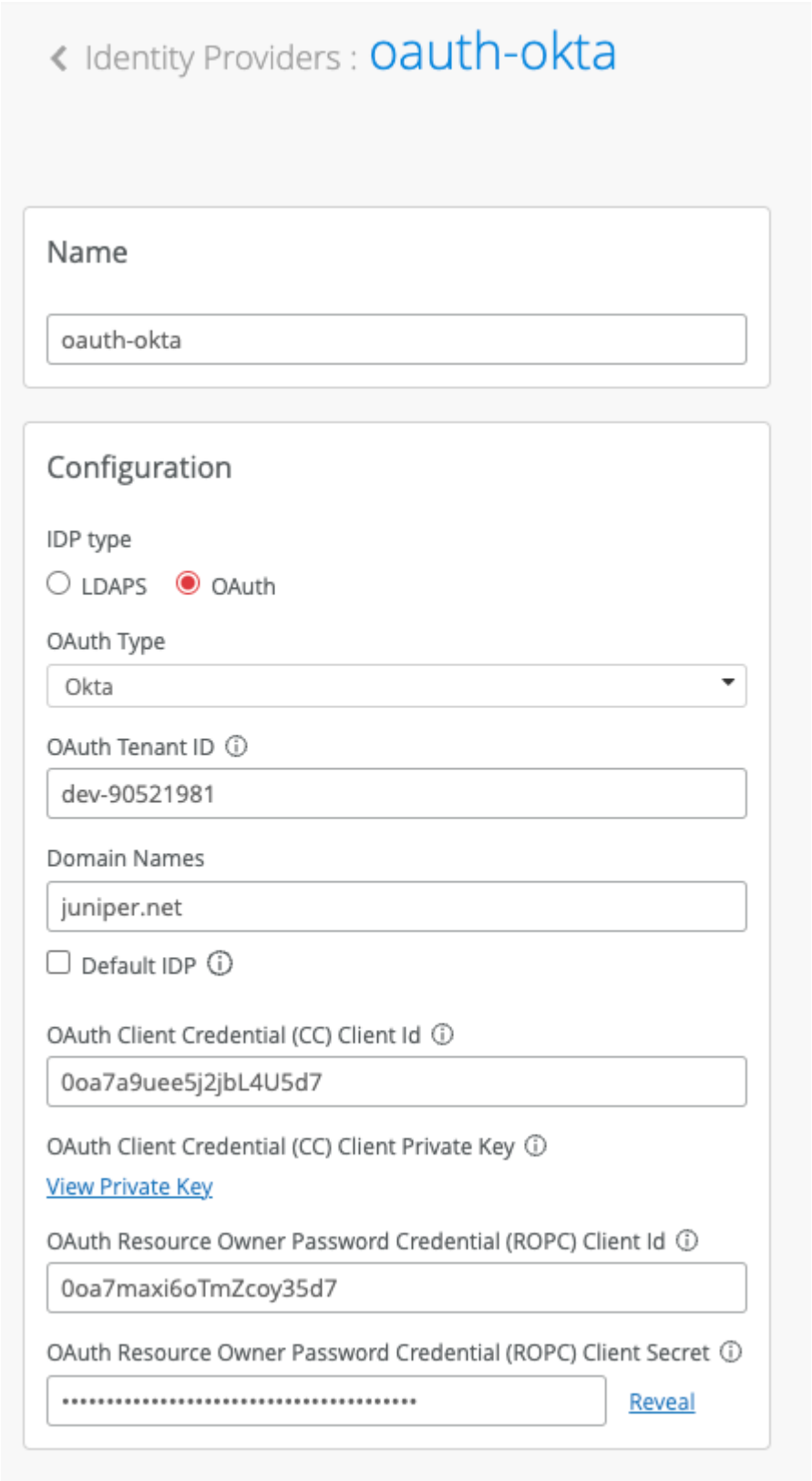
```
<Private key you copied Part 3 /
Step 2>
```

- OAuth ROPC Client ID=

```
<Client ID from Part 2 / Step 3
of this guide>
```

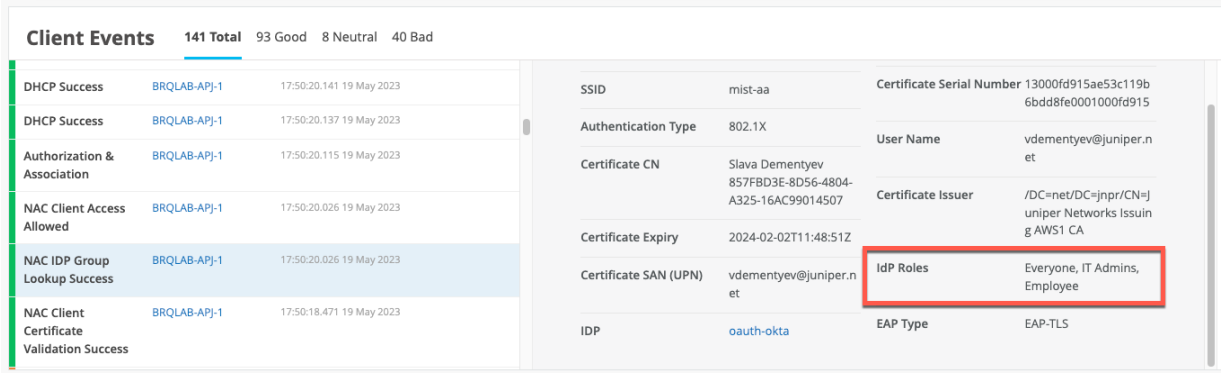
- OAuth ROPC Client Secret=

```
<Secret from Part 2 / Step 3
of this guide>
```

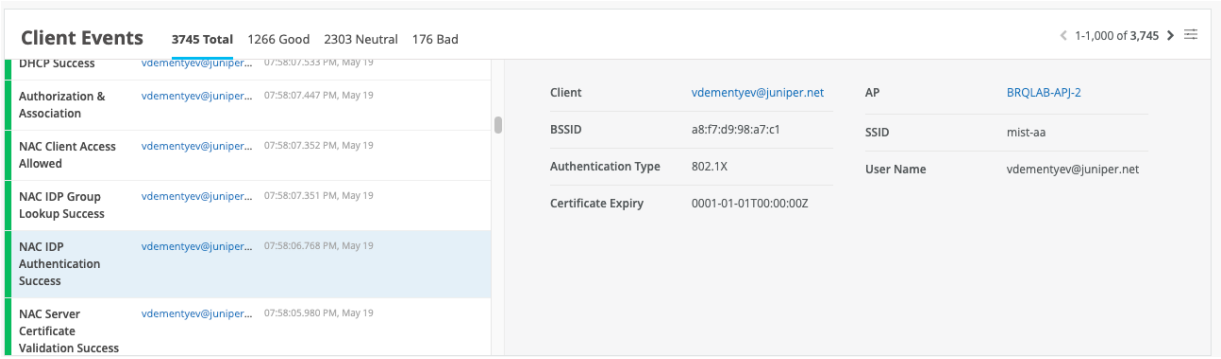

Identity Providers : oauth-okta
Name
oauth-okta
Configuration
IDP type
LDAPS
OAuth
OAuth Type
Okta
OAuth Tenant ID
dev-90521981
Domain Names
juniper.net
Default IDP
OAuth Client Credential (CC) Client Id
0oa7a9uee5j2jbL4U5d7
OAuth Client Credential (CC) Client Private Key
View Private Key
OAuth Resource Owner Password Credential (ROPC) Client Id
0oa7maxi6oTmZcoy35d7
OAuth Resource Owner Password Credential (ROPC) Client Secret
Reveal

Validation

In the case of a successful EAP-TLS client authentication with a subsequent IdP lookup against Okta, you should see a **NAC Group Lookup Success** event with IdP roles fetched from Okta.



In the case of a successful EAP-TTLS authentication, you would also see a successful **NAC IDP Authentication Success** event indicating that Okta has verified the user credentials.



Own LDAP Directory

The steps below are provided to give you an idea about the steps taken when you want to test against a simple minimalistic LDAP repository. The LDAP repository and its directory structure were kept as basic as possible. We suggest that you not put it into production, but it might help to understand the processes if you intend to create a similar integration.

NOTE: The information presented in this chapter is limited to lab usage only and not meant to be used in production.

Own LDAP Directory Part 1: Create an Instance with a Public IP Address

The first step is to set up an instance running somewhere that can host the LDAP software. This can be a container, a virtual machine or a physical server. You do not require much in the way of resources or CPU power. You can even run the service on a Raspberry Pi. An important aspect of this solution is that you need to have a static public IP address and translate the LDAPS TCP port 636 to the private LDAP instance if the instance is not directly connected to the static public IP address.

Remember that the Juniper Mist authentication cloud (located on the Internet) needs to be able to open a connection to the private LDAP instance to check credentials. If you host that instance locally, you likely need to ask your IT team to allow this connection through an enterprise firewall. An alternative is to create an instance in a public cloud provider as you can then more easily get a public IP address mapped to that instance.

Whichever option you choose, note down the public IP address for the LDAP instance for the next process. In our example, we created a small Ubuntu 22.04 VM on a public cloud to run the LDAP instance.

Own LDAP Directory Part 2: Create Certificates for LDAPS

Once the LDAP instance with a static public IP address is in place, you must create a certificate using the customer's PKI for a TLS server. If you plan to put the service into production later, the best option is to add it to DNS with FQDN. Here, we just use the static IP address for simplicity. The example below uses a home-grown PKI based on OpenSSL, so you likely need to use similar steps that are relevant to the customer's PKI. Obviously, you need to substitute 127.0.0.1 with your own IP address.

```
cd examplePKI
openssl genrsa -out ldapserveraz.example.key 4096
openssl req -new -key ldapserveraz.example.key -subj '/C=US/ST=California/L=Sunnyvale/O=Example
TEST-Corp./OU=IT-Department/CN=127.0.0.1' -out ldapserveraz.example.csr
cat <<EOF >ca-server.extensions.cnf
basicConstraints=critical,CA:FALSE
keyUsage = digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment,keyAgreement
extendedKeyUsage = serverAuth
subjectAltName=IP:127.0.0.1
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer
EOF
openssl ca -config casign.conf -extfile ca-server.extensions.cnf -out ldapserveraz.example.pem -
infiles ldapserveraz.example.csr
openssl x509 -in ldapserveraz.example.pem -out strip.pem
cat strip.pem >ldapserveraz.example.pem
cat ldapserveraz.example.pem
-----BEGIN CERTIFICATE-----
.
.
-----END CERTIFICATE-----
cat ldapserveraz.example.key
-----BEGIN RSA PRIVATE KEY-----
.
```

```
.
-----END RSA PRIVATE KEY-----
```

Own LDAP Directory Part 3: Install the LDAP Service

In our example, we used a Docker container inside our VM, so the first step was installing Docker itself:

```
apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common -
y
apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 7EA0A9C3F273FCD8
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs)
stable"
apt-get update
apt-get install docker-ce docker-ce-cli containerd.io docker-compose -y
```

Then, you can install an OpenLDAP Docker container and configure the certificates for LDAPS:

```
# create some storage directories
docker rm -f openldap-server
rm -fR /data/slapd/*
mkdir -p /data/slapd/config
mkdir /data/slapd/database
mkdir /data/slapd/certs
chmod 775 -R /data/slapd
chown -R $USER:docker /data/slapd
apt-get install -y ldap-tools
# we use the cacert.pem File of our root-ca
# as our Enterprise root CA
cat <<EOF >/data/slapd/certs/cacert.pem
-----BEGIN CERTIFICATE-----
.
.
-----END CERTIFICATE-----
EOF
cat <<EOF >/data/slapd/certs/ldapserver.example.pem
-----BEGIN CERTIFICATE-----
.
# use the server certificate generated above for the cn=<public-ip>
.
-----END CERTIFICATE-----
EOF
```

```
cat <<EOF >/data/slapd/certs/ldapserver.example.key
-----BEGIN RSA PRIVATE KEY-----
.
# use the server key generated above for the cn=<public-ip>
.
-----END RSA PRIVATE KEY-----
EOF
# create and remember a strong ADMIN password
ADMINPASS=`cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 20 | head -n 1`
echo 'The Admin password is:'$ADMINPASS >ldapadminpass.txt
echo 'The Admin password is:'$ADMINPASS
The Admin password is:MC0xxxxxxxxxxxxRjQ6IC
# lauch the ldap docker instance
docker rm -f openldap-server
docker run \
  --restart=always \
  --name openldap-server \
  -p 389:389 \
  -p 636:636 \
  --env LDAP_ORGANISATION="example.net" \
  --env LDAP_DOMAIN="ldap.example.net" \
  --env LDAP_BASE_DN="dc=ldap,dc=example,dc=net" \
  --env LDAP_ADMIN_PASSWORD=$ADMINPASS \
  --volume /data/slapd/database:/var/lib/ldap \
  --volume /data/slapd/config:/etc/ldap/slapd.d \
  --volume /data/slapd/certs:/container/service/slapd/assets/certs \
  --env LDAP_READONLY_USER=true \
  --env LDAP_READONLY_USER_USERNAME=readonly \
  --env LDAP_READONLY_USER_PASSWORD=juniper123 \
  --env LDAP_TLS=true \
  --env LDAP_TLS_CRT_FILENAME=ldapserver.example.pem \
  --env LDAP_TLS_KEY_FILENAME=ldapserver.example.key \
  --env LDAP_TLS_CA_CRT_FILENAME=cacert.pem \
  --env LDAP_TLS_VERIFY_CLIENT=try \
  --detach osixia/openldap:latest
#docker rm -f phpldapadmin
#docker run \
    --restart=always \
    --name phpldapadmin \
    -p 80:80 \
    --env PHPLDAPADMIN_LDAP_HOSTS=172.31.0.10 \
```

```
    --env PHPLDAPADMIN_HTTPS='false' \
    --detach osixia/phpldapadmin:latest
```

To test your service installation, you can use the following:

```
# check if ldap docker works at all (unsecure as readonly user)
ldapsearch \
               -h 127.0.0.1 \
               -p 389 \
               -D 'cn=readonly,dc=ldap,dc=example,dc=net' \
               -w 'juniper123' \
               -b 'dc=ldap,dc=example,dc=net' \
               '(objectClass=*)'
# extended LDIF
#
# LDAPv3
# base <dc=ldap,dc=example,dc=net> with scope subtree
# filter: (objectClass=*)
# requesting: ALL
#
# ldap.example.net
dn: dc=ldap,dc=example,dc=net
objectClass: top
objectClass: dcObject
objectClass: organization
o: example.net
dc: ldap
# search result
search: 2
result: 0 Success
# numResponses: 2
# numEntries: 1
```

Own LDAP Directory Part 4: Create a Basic LDAP Directory

To create a small LDAP directory for testing, we imported a small one that was created elsewhere as an LDIF file.

```
# create a small ldif file for import
cat <<EOF >import.ldif
# LDIF Export for dc=ldap,dc=example,dc=net
# Server: 192.168.10.14 (192.168.10.14)
```

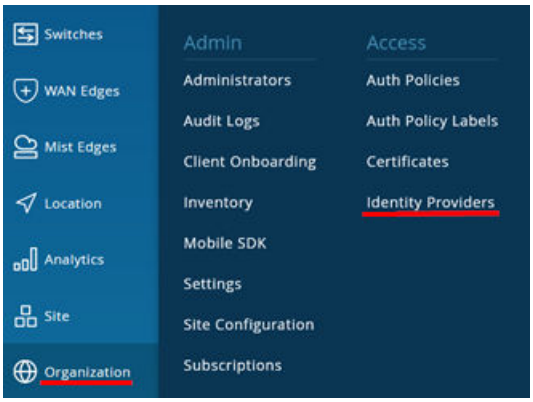
```
# Search Scope: sub
# Search Filter: (objectClass=*)
# Total Entries: 8
#
# Generated by phpLDAPadmin (http://phpldapadmin.sourceforge.net) on June 23, 2023 2:55 pm
# Version: 1.2.5
version: 1
# Entry 1: dc=ldap,dc=example,dc=net
#dn: dc=ldap,dc=example,dc=net
#dc: ldap
#o: example.net
#objectclass: top
#objectclass: dcObject
#objectclass: organization
# Entry 2: ou=groups,dc=ldap,dc=example,dc=net
dn: ou=groups,dc=ldap,dc=example,dc=net
objectclass: organizationalUnit
objectclass: top
ou: groups
# Entry 3: cn=employees,ou=groups,dc=ldap,dc=example,dc=net
dn: cn=employees,ou=groups,dc=ldap,dc=example,dc=net
cn: employees
gidnumber: 501
memberuid: user01
memberuid: user02
memberuid: user03
objectclass: posixGroup
objectclass: top
# Entry 4: cn=employer,ou=groups,dc=ldap,dc=example,dc=net
dn: cn=employer,ou=groups,dc=ldap,dc=example,dc=net
cn: employer
gidnumber: 500
objectclass: posixGroup
objectclass: top
# Entry 5: ou=users,dc=ldap,dc=example,dc=net
dn: ou=users,dc=ldap,dc=example,dc=net
objectclass: organizationalUnit
objectclass: top
ou: users
# Entry 6: cn=user01,ou=users,dc=ldap,dc=example,dc=net
dn: cn=user01,ou=users,dc=ldap,dc=example,dc=net
cn: user01
gidnumber: 501
```

```
homedirectory: /home/users/user01
loginshell: /bin/bash
mail: user01@example.net
objectclass: inetOrgPerson
objectclass: posixAccount
objectclass: top
sn: user01
uid: user01
uidnumber: 1000
userpassword: juniper123
# Entry 7: cn=user02,ou=users,dc=ldap,dc=example,dc=net
dn: cn=user02,ou=users,dc=ldap,dc=example,dc=net
cn: user02
gidnumber: 501
homedirectory: /home/users/user02
loginshell: /bin/bash
mail: user02@example.net
objectclass: inetOrgPerson
objectclass: posixAccount
objectclass: top
sn: user02
uid: user02
uidnumber: 1001
userpassword: juniper123
# Entry 8: cn=user03,ou=users,dc=ldap,dc=example,dc=net
dn: cn=user03,ou=users,dc=ldap,dc=example,dc=net
cn: user03
gidnumber: 501
homedirectory: /home/users/user03
mail: user03@example.net
objectclass: inetOrgPerson
objectclass: posixAccount
objectclass: top
sn: user03
uid: user03
uidnumber: 1002
userpassword: juniper123
EOF
# import your LDIF file to create the accounts
ldapadd \
               -h 127.0.0.1 \
               -p 389 \
               -D 'cn=admin,dc=ldap,dc=example,dc=net' \
```

```
                -w $ADMINPASS \
                -f import.ldif
adding new entry "ou=groups,dc=ldap,dc=example,dc=net"
adding new entry "cn=employees,ou=groups,dc=ldap,dc=example,dc=net"
adding new entry "cn=employer,ou=groups,dc=ldap,dc=example,dc=net"
adding new entry "ou=users,dc=ldap,dc=example,dc=net"
adding new entry "cn=user01,ou=users,dc=ldap,dc=example,dc=net"
adding new entry "cn=user02,ou=users,dc=ldap,dc=example,dc=net"
adding new entry "cn=user03,ou=users,dc=ldap,dc=example,dc=net"
```

Own LDAP Directory Part 5: Juniper Mist Cloud Configuration

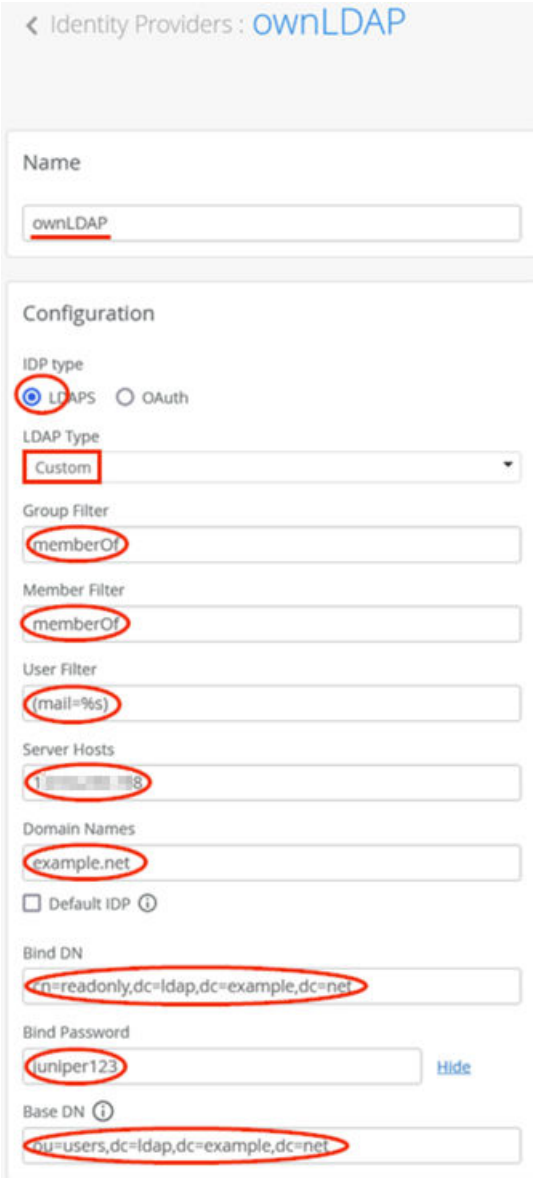
Navigate to **Organization > Identity Providers**.



Add a new IdP with the following configuration:

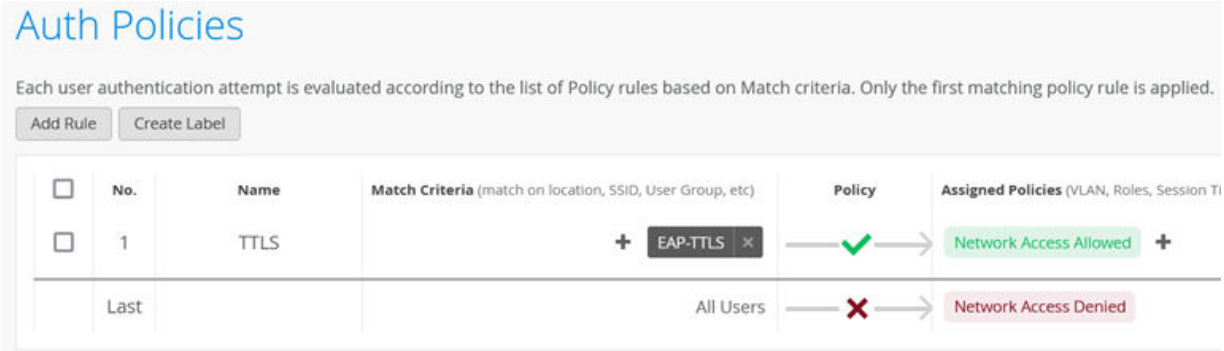
- Name=`ownLDAP`
- IDP Type=`LDAPS`
- LDAP Type=`Custom`
- Group Filter=`memberOf`
- Member Filter=`memberOf`
- User Filter=`(mail=%s)` (you may also use `(uid=%s)` but then the TTLS client must only use their name and not their email address when authenticating)
- Server Hosts=`<DNS-FQDN OR Public-IP>` (Note that you MUST have a matching server certificate on the LDAPS server where the common name matches what you configure here)
- Domain Names=`example.net`
- Bind DN=`cn=readonly,dc=ldap,dc=example,dc=net`

- Bind Password=`juniper123`
- Base DN=`ou=users,dc=ldap,dc=example,dc=net`



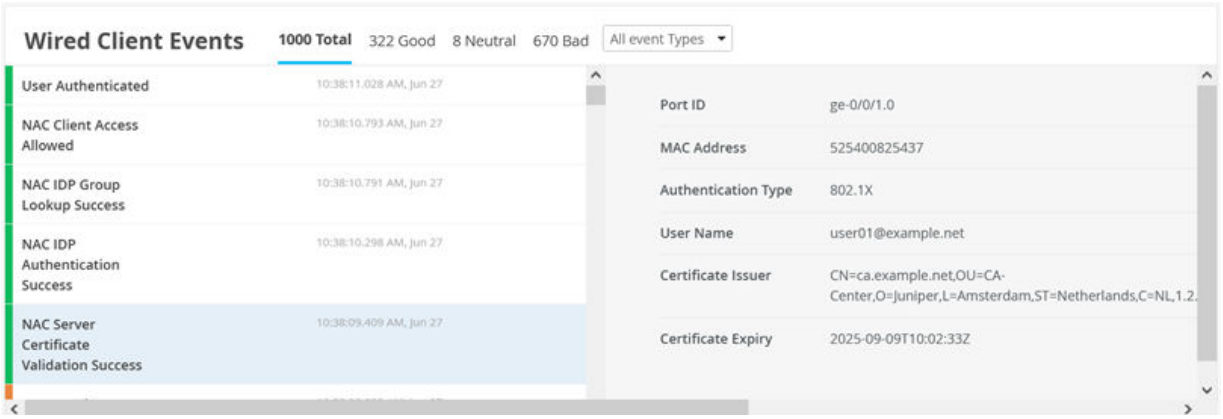
In our case, the certificate for the LDAPS connection was made from the same customer PKI. Hence, no additional steps are required. Should it be from an external LDAPS service provider, you need to obtain and add a client certificate. See the next figure:

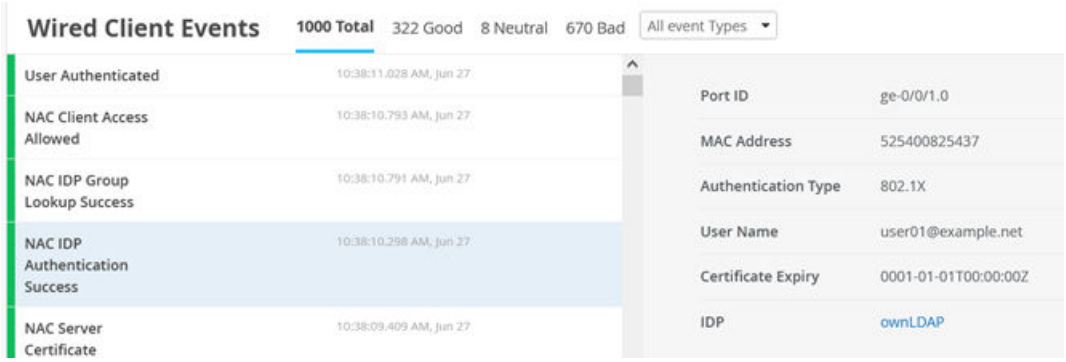
To test the IdP integration under **Organization > Auth Policies**, create a simple EAP-TTLS rule like that shown below:

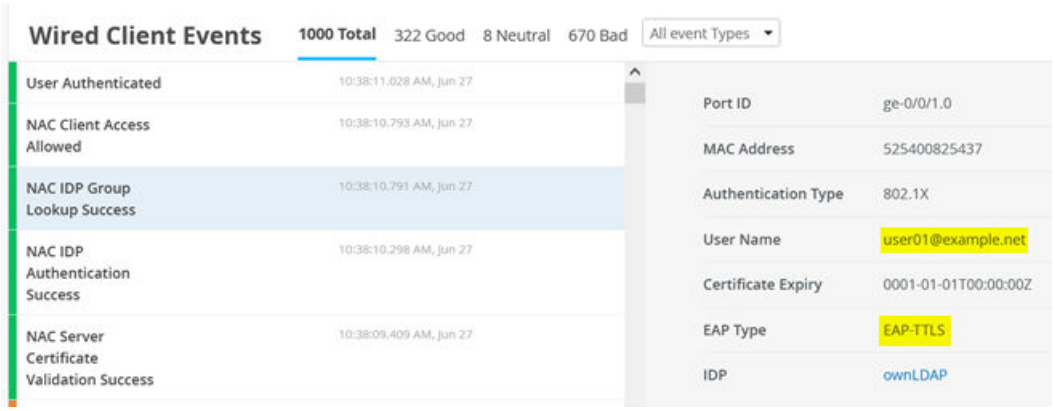


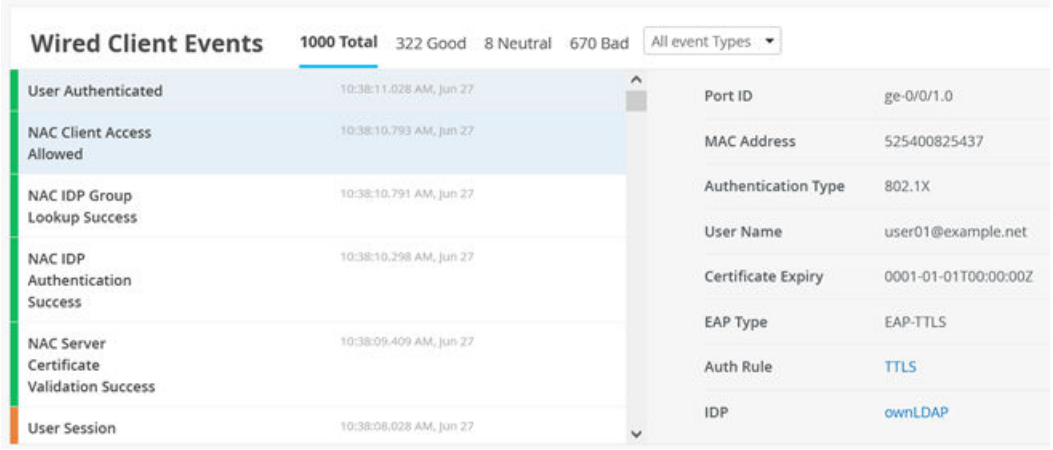
In our directory, the EAP supplicant can then use the 3 usernames (`user01@example.net` , `user01@example.net` , and `user03@example.net`) with the password `juniper123` for all 3 usernames (or you can change this in the LDIF file before importing).

When going through the authentication process, you should see the following events.




Wired Client Events
1000 Total
322 Good
8 Neutral
670 Bad
All event Types
User Authenticated
NAC Client Access Allowed
NAC IDP Group Lookup Success
NAC IDP Authentication Success
NAC Server Certificate
Port ID
ge-0/0/1.0
MAC Address
525400825437
Authentication Type
802.1X
User Name
user01@example.net
Certificate Expiry
0001-01-01T00:00:00Z
IDP
ownLDAP


Wired Client Events
1000 Total
322 Good
8 Neutral
670 Bad
All event Types
User Authenticated
NAC Client Access Allowed
NAC IDP Group Lookup Success
NAC IDP Authentication Success
NAC Server Certificate Validation Success
Port ID
ge-0/0/1.0
MAC Address
525400825437
Authentication Type
802.1X
User Name
user01@example.net
Certificate Expiry
0001-01-01T00:00:00Z
EAP Type
EAP-TTLS
IDP
ownLDAP


Wired Client Events
1000 Total
322 Good
8 Neutral
670 Bad
All event Types
User Authenticated
NAC Client Access Allowed
NAC IDP Group Lookup Success
NAC IDP Authentication Success
NAC Server Certificate Validation Success
User Session
Port ID
ge-0/0/1.0
MAC Address
525400825437
Authentication Type
802.1X
User Name
user01@example.net
Certificate Expiry
0001-01-01T00:00:00Z
EAP Type
EAP-TTLS
Auth Rule
TTLS
IDP
ownLDAP

Appendix: Test Cases to Be Performed

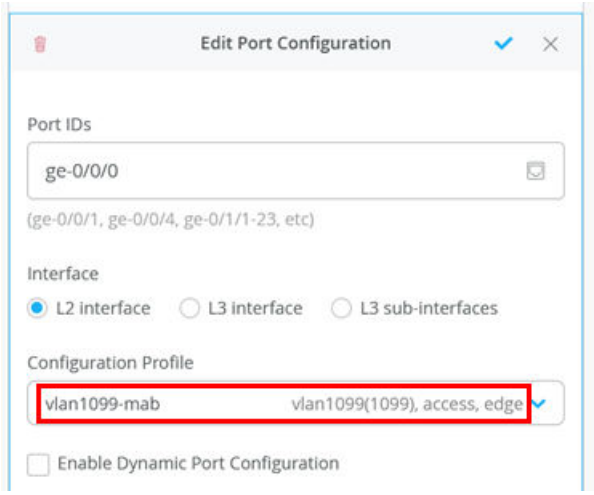
IN THIS SECTION

In this chapter, we are sharing information about the major test cases performed for this JVD and how you can repeat and review them in your own environment.

Authentication MAB Wired Client

To test MAC address-based authentication of a wired client, execute the following steps one by one.

First, we need to configure the port on the access switch where the wired client is attached to use the profile for MAB that we defined in the switch template in Figure 25 on page 83. Change the configuration profile to `vlan1099-mab`.



NOTE: After this is applied, your wired clients will no longer be able to communicate with the network since we have not authenticated them yet.

(Optional) Remote Shell to the switch to review the configurations applied for RadSec, the certificate, and the port.

```
mist@switch1> show configuration | display set | match dot1x
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37400
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set protocols dot1x authenticator authentication-profile-name dot1x
set protocols dot1x authenticator interface vlan1099-mab supplicant multiple
set protocols dot1x authenticator interface vlan1099-mab mac-radius restrict
set protocols dot1x authenticator interface vlan1099-mab mac-radius authentication-protocol pap
mist@switch1> show configuration | display set | match vlan1099-mab | match interfaces
set groups vlan1099-mab interfaces <*> unit 0 family ethernet-switching vlan members vlan1099
set interfaces interface-range vlan1099-mab member ge-0/0/0
set interfaces interface-range vlan1099-mab apply-groups vlan1099-mab
mist@switch1> show configuration | display set | match access
set groups top access radius-server 3.33.153.159 secret "$9$7HdYoJZj.mTGD.5F3tp"
```

```
set groups top access radius-server 3.33.153.159 timeout 5
set groups top access radius-server 3.33.153.159 radsec-destination 895
set groups top access radius-server 15.197.139.214 secret "$9$7HdYoJZj.mTGD.5F3tp"
set groups top access radius-server 15.197.139.214 timeout 5
set groups top access radius-server 15.197.139.214 radsec-destination 896
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37400
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set groups top access radsec destination 895 address 3.33.153.159
set groups top access radsec destination 895 port 2083
set groups top access radsec destination 895 tls-certificate mist-nac-device-cert
set groups top access radsec destination 895 tls-force-ciphers low
set groups top access radsec destination 895 tls-min-version v1.2
set groups top access radsec destination 895 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 895 tls-timeout 30
set groups top access radsec destination 896 address 15.197.139.214
set groups top access radsec destination 896 port 2083
set groups top access radsec destination 896 tls-certificate mist-nac-device-cert
set groups top access radsec destination 896 tls-force-ciphers low
set groups top access radsec destination 896 tls-min-version v1.2
set groups top access radsec destination 896 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 896 tls-timeout 30
mist@switch1> show security pki ca-certificate
LSYS: root-logical-system
  CA profile: mist-vpn-ca
Certificate identifier: mist-vpn-ca
  Issued to: cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae, Issued by: C = US, O = Mist, OU = OrgCA, CN =
cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-28-2023 09:37 UTC
    Not after: 08-25-2033 09:37 UTC
  Public key algorithm: rsaEncryption(4096 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show security pki local-certificate
```

```
LSYS: root-logical-system
Certificate identifier: mist-nac-device-cert
  Issued to: 00cc34f37400, Issued by: C = US, O = Mist, OU = OrgCA, CN = cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-23-2024 10:11 UTC
    Not after: 08-23-2025 10:11 UTC
  Public key algorithm: rsaEncryption(2048 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show system connections | match 2083
tcp4       0      0  10.33.33.19.59751
15.197.139.214.2083                    ESTABLISHED
tcp4       0      0  10.33.33.19.65050
3.33.153.159.2083                      ESTABLISHED
mist@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/0.0    Authenticator  Connecting
mist@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Multiple
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Enabled
  Mac Radius Restrict: Enabled
  Mac Radius Authentication Protocol: PAP
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
  Supplicant timeout: 30 seconds
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 0
```

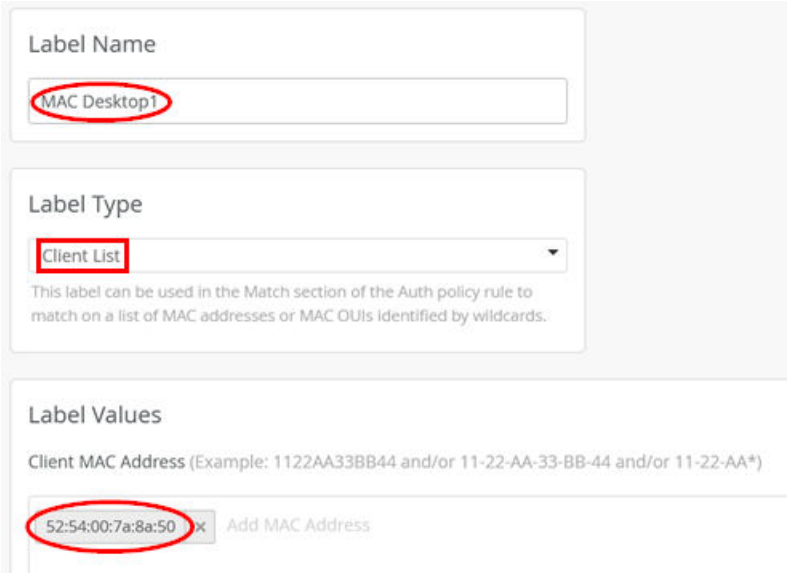
You must know the MAC address you want to authenticate. The following example shows the retrieval of this information from a Linux client.

```
root@desktop1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
```

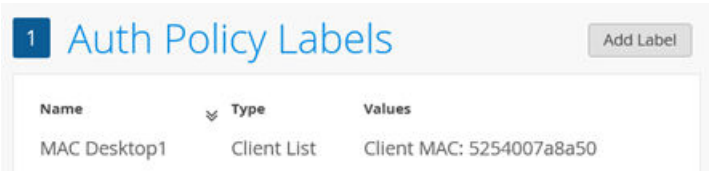
```
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
.
4: ens5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
1000
    link/ether 52:54:00:7a:8a:50 brd ff:ff:ff:ff:ff:ff
    inet 10.99.99.99/24 brd 10.99.99.255 scope global ens5
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe7a:8a50/64 scope link
       valid_lft forever preferred_lft forever
```

Next, go to **Organization > Auth Policy Labels** and create a label identifying this MAC address:

- Label Name=`MAC Desktop1`
- Label Type=`Client List`
- Label Values=`<your-MAC>`

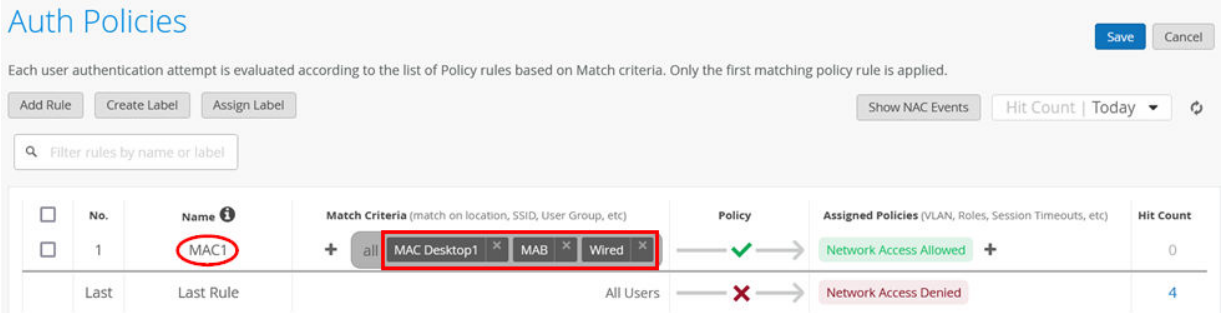


You should only see this label right now.



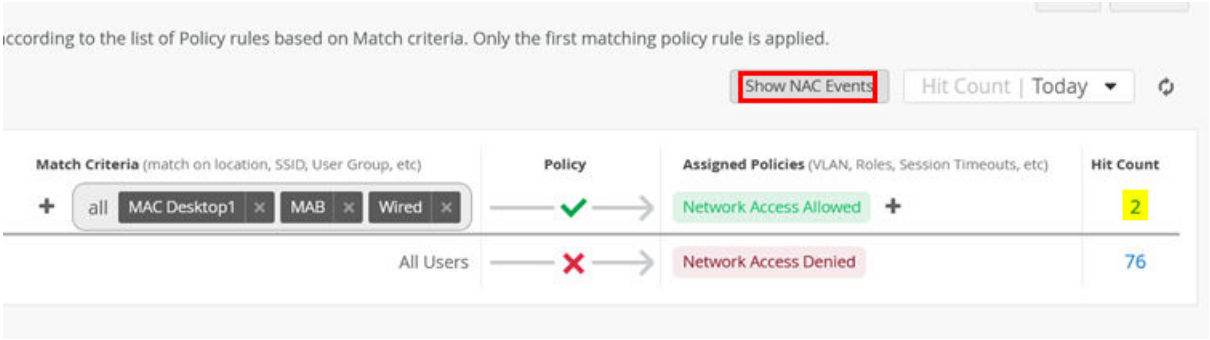
Then, go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`MAC1`
- Match Criteria=`MAC Desktop1` and `MAB` and (optional) `Wired`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`

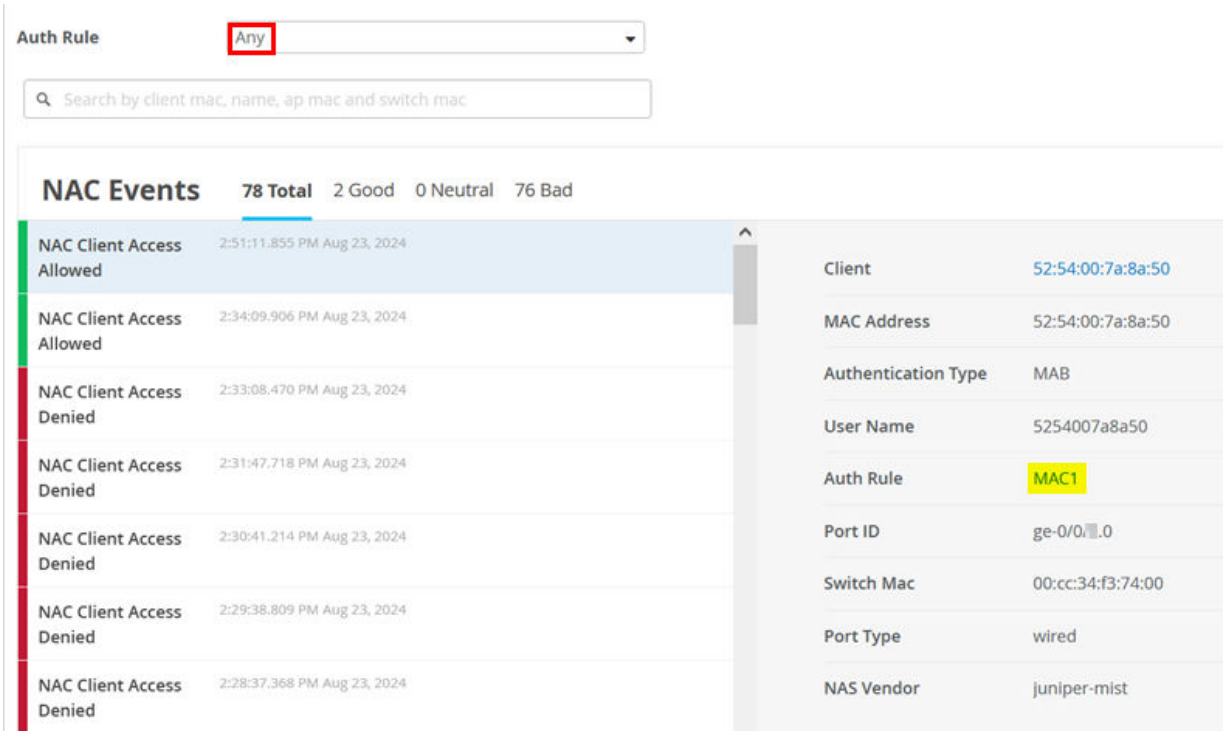


NOTE: The session reauthentication interval for MAC addresses is set to 10 minutes by default. If you do not change this interval value using additional CLI configuration and a MAC address is not initially authenticated, it can take up to 10 minutes to get a successful MAC authentication.

You can confirm the success of your authentication policy in this window when it increments the **Hit Count** value. Also, select **Show NAC Events**.

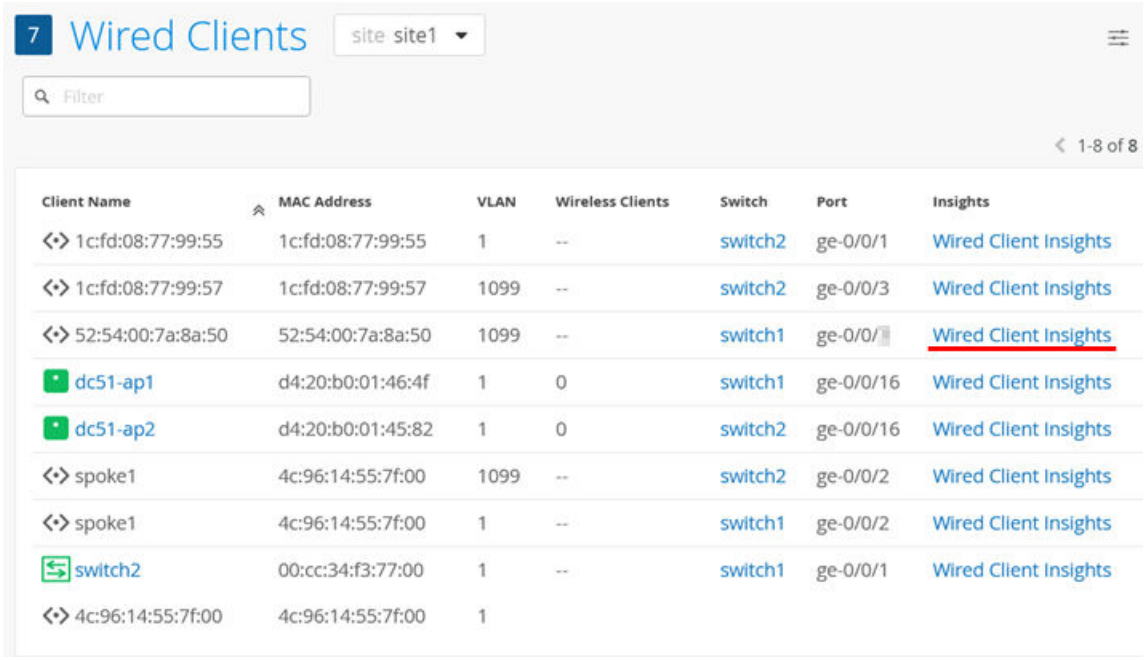


You can see the information about your client here:

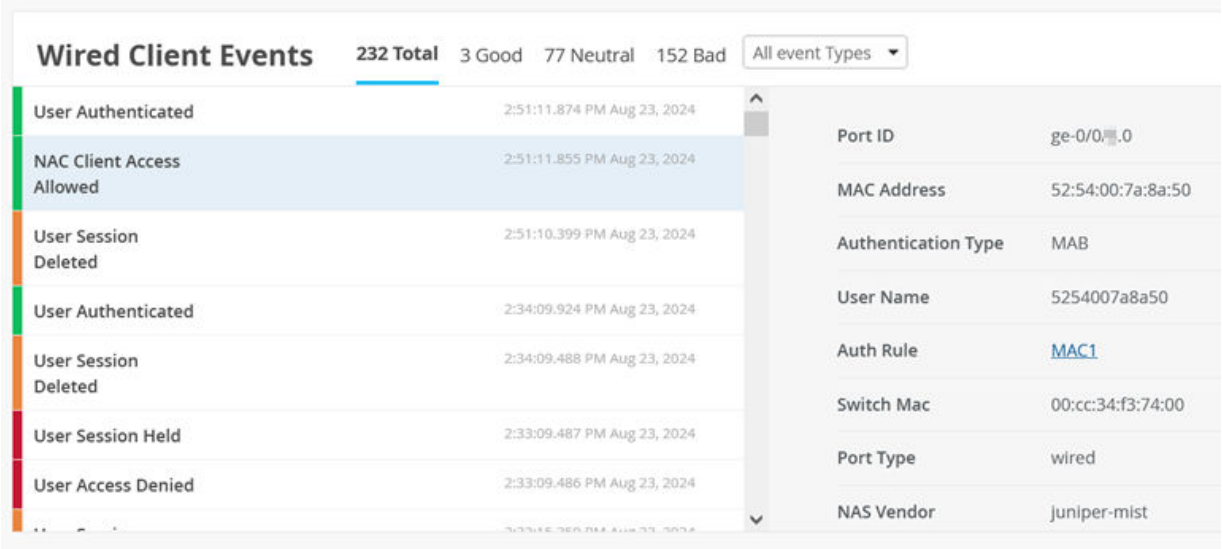


You can also go to **Clients > Wired Clients**.

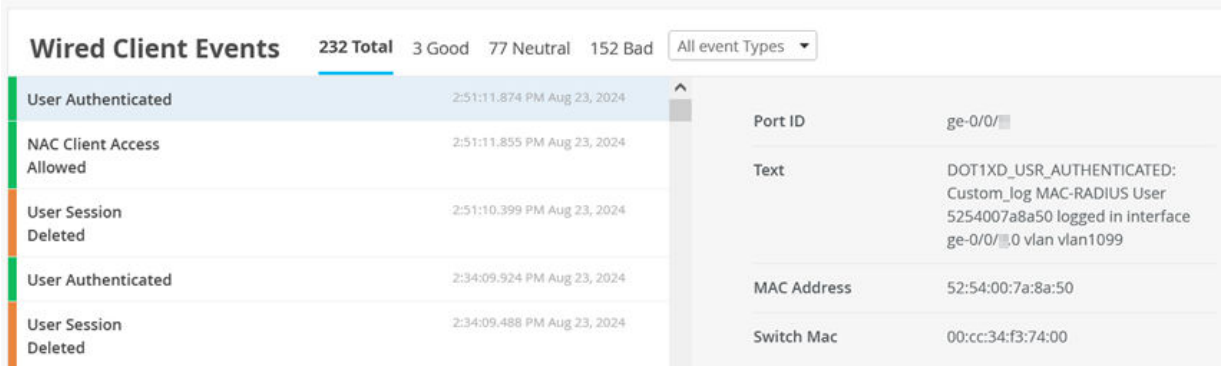
Then, find your client in the list and click **Wired Client Insights**.



Then, you can inspect the **Wired Client Events** for **NAC Client Access Allowed** events.



Also see **User Authenticated** events.



(Optional) You can Remote Shell to the switch and run the commands shown below:

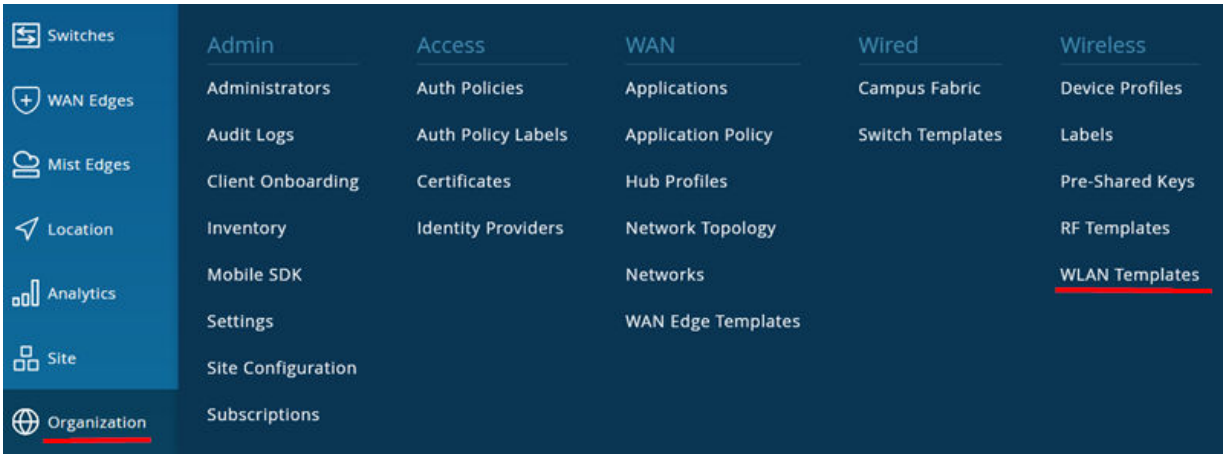
```
root@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     State          MAC address         Method            Vlan  User
ge-0/0/0.0    Authenticated  52:54:00:7A:8A:50   Mac Radius        1099  5254007a8a50
root@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Multiple
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
```

```
Mac Radius: Enabled
Mac Radius Restrict: Enabled
Mac Radius Authentication Protocol: PAP
Reauthentication: Enabled
Reauthentication interval: 3600 seconds
Supplicant timeout: 30 seconds
Server timeout: 30 seconds
Maximum EAPOL requests: 2
Guest VLAN member: not configured
Number of connected supplicants: 1
  Supplicant: 5254007a8a50, 52:54:00:7A:8A:50
    Operational state: Authenticated
    Backend Authentication state: Idle
    Authentication method: Mac Radius
    Authenticated VLAN: vlan1099
    Session Reauth interval: 3600 seconds
    Reauthentication due in 3483 seconds
    Session Accounting Interim  Interval: 36000 seconds
    Accounting Update  due in 35883 seconds
    Eapol-Block: Not In Effect
    Domain: Data
```

Authentication MAB Wireless Client

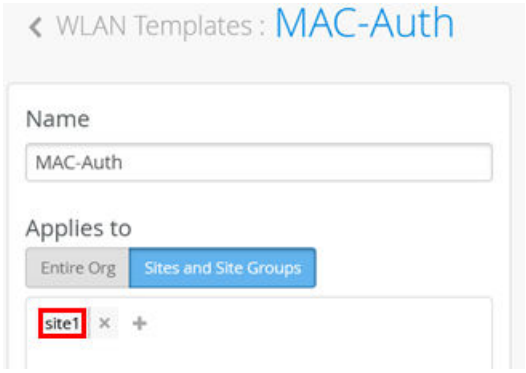
In this example, we use MAC address-based authentication for wireless clients. We combine it with PSK authentication to achieve some minimal traffic encryption over the air as MAC addresses are easy to spot and mimic by a potential attacker.

An SSID for wireless can be configured in several ways. In our example, we use a WLAN template by first navigating to **Organization > WLAN Templates**.

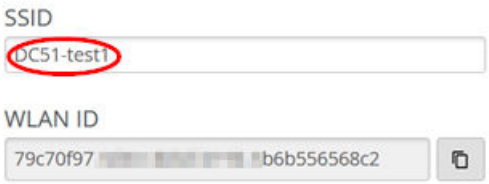


Create a new template with the following settings:

- Name=`MAC-Auth`
- Applies to:
  - Sites and Site Groups=`site1`

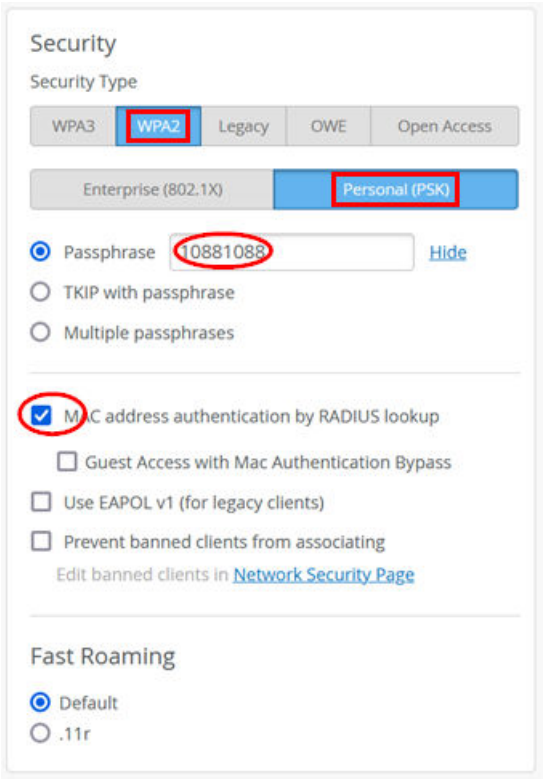


Create an SSID similar to the figure shown below:



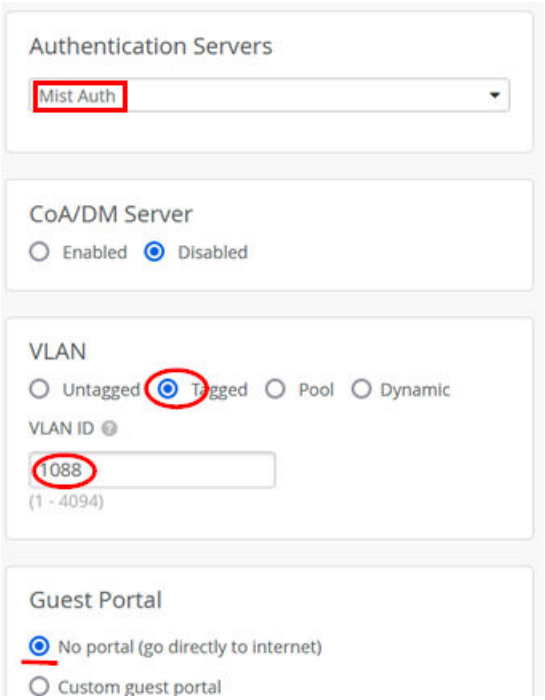
Under **Security**, configure the following:

- Security Type=`WPA2` and `Personal (PSK)`
- Passphrase=`10881088` (or anything else you remember)
- MAC address authentication by RADIUS lookup=`Checked`. **This is important for our test!**

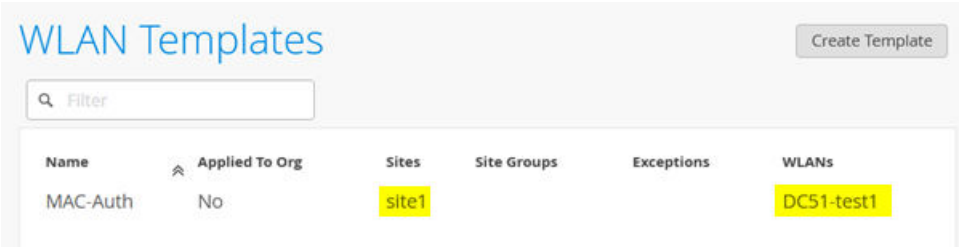


Then, configure the following settings:

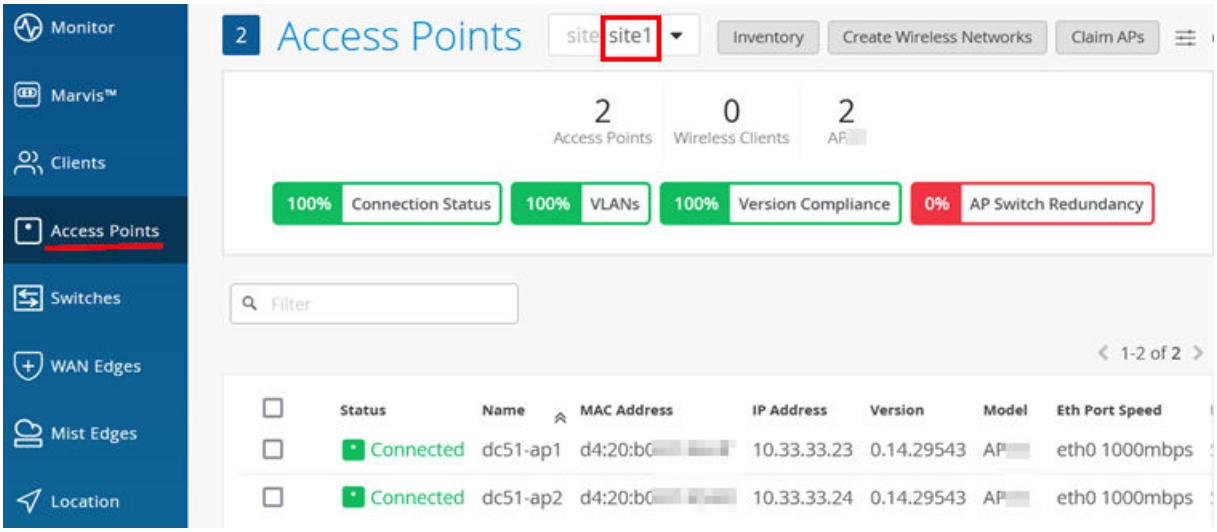
- Authentication Servers=`Mist Auth`
- VLAN=`Tagged`
- VLAN ID=`1088`



After saving the template, you should see the following configuration:



Next, go **Access Points** and select your site to review the APs. Select one AP.



Review the AP configuration applied and ensure the SSID from the template appears in the WLANs tab.

The next step is to determine a wireless client's MAC address and allow it to use this SSID. There are several ways to do this, and they are different for every client OS. The below example shows how to retrieve the wireless interfaces available on a Linux client and use that information to find the MAC address of that client:

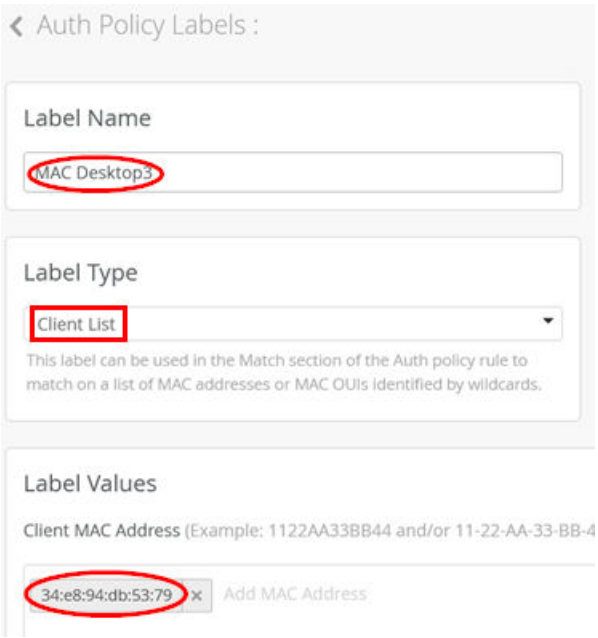
```
root@desktop3:~# iwconfig
ens3      no wireless extensions.
wlx34e894db5379  unassociated  Nickname:"WIFI@RTL8821AU"
          Mode:Managed  Frequency=2.412 GHz  Access Point: Not-Associated
          Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
lo        no wireless extensions.
root@desktop3:~# ip a
.
.
3: wlx34e894db5379: <NO-CARRIER,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state DORMANT
group default qlen 1000
    link/ether 34:e8:94:db:53:79 brd ff:ff:ff:ff:ff:ff
```

NOTE: A mobile client OS may frequently change the Wi-Fi adapter's MAC address, making it impossible to manage the device using its MAC address. Sometimes this option can be disabled, but you need to know how to change this configuration on the device.

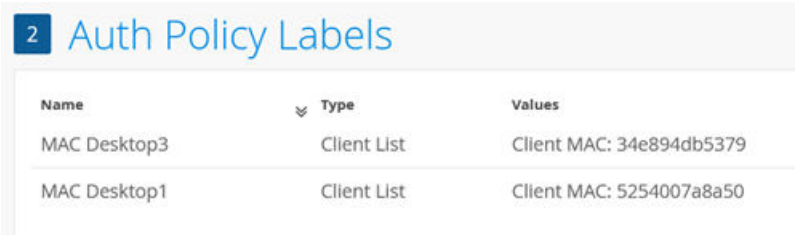
Next, we need to specify a label that identifies this MAC address by navigating to **Organization > Auth Policy Labels** and creating the following label identifying this MAC address:

- Label Name=`MAC Desktop3`

- Label Type=`Client List`
- Label Values=`<your-MAC>`

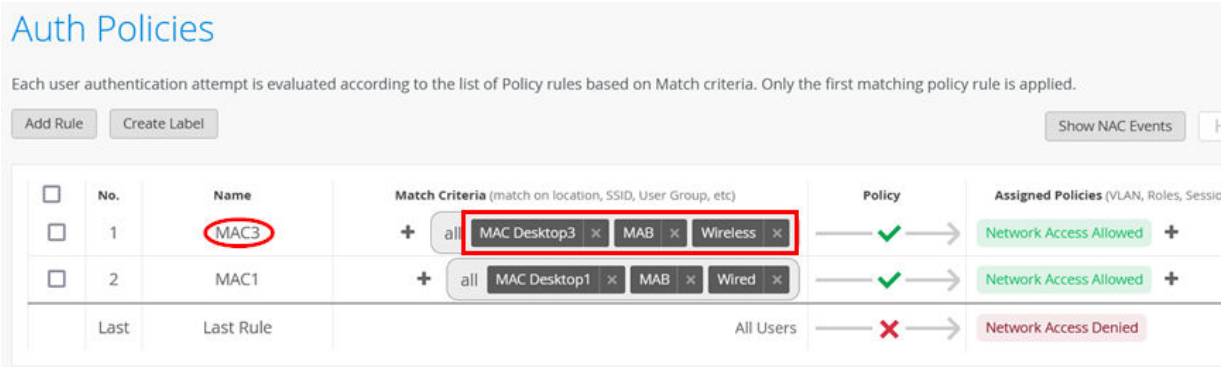


You should now have two MAC address labels.



Then, go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`MAC3`
- Match Criteria=`MAC Desktop3` and `MAB` and (optional) `Wireless`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



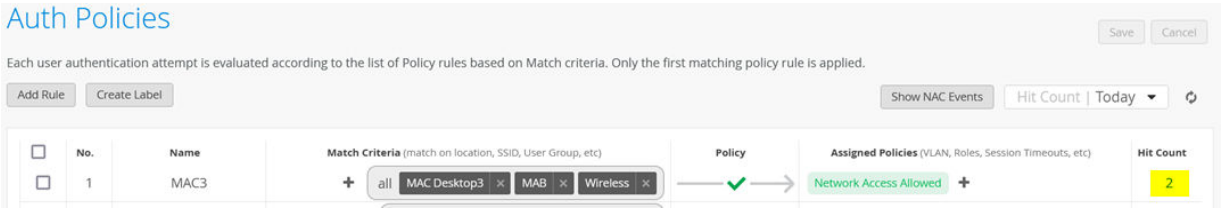
After saving the new ruleset, you are ready to have the wireless client use the configured SSID and attach to the AP. Again, this example shows a Linux client using the wpa_supplicant:

```
# write a wpa supplicant configuration file
cat <<EOF >/etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant
ctrl_interface_group=wheel
ap_scan=1
network={
        ssid="DC51-test1"
        psk="10881088"
}
EOF
# run the supplicant in foreground so that we can see its debugging messages
root@desktop3:~# rm -f /var/run/wpa_supplicant/wlx34e894db5379
root@desktop3:~# wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -D nl80211 -i
wlx34e894db5379
.
Successfully initialized wpa_supplicant
wlx34e894db5379: Trying to associate with d4:20:b0:0c:a7:d4 (SSID='DC51-test1' freq=5260 MHz)
wlx34e894db5379: CTRL-EVENT-STARTED-CHANNEL-SWITCH freq=5260 ht_enabled=1 ch_offset=1
ch_width=40 MHz cf1=5270 cf2=0
wlx34e894db5379: Associated with d4:20:b0:0c:a7:d4
wlx34e894db5379: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlx34e894db5379: CTRL-EVENT-REGDOM-CHANGE init=COUNTRY_IE type=COUNTRY alpha2=US
wlx34e894db5379: WPA: Key negotiation completed with d4:20:b0:0c:a7:d4 [PTK=CCMP GTK=CCMP]
wlx34e894db5379: CTRL-EVENT-CONNECTED - Connection to d4:20:b0:0c:a7:d4 completed [id=0 id_str=]
```

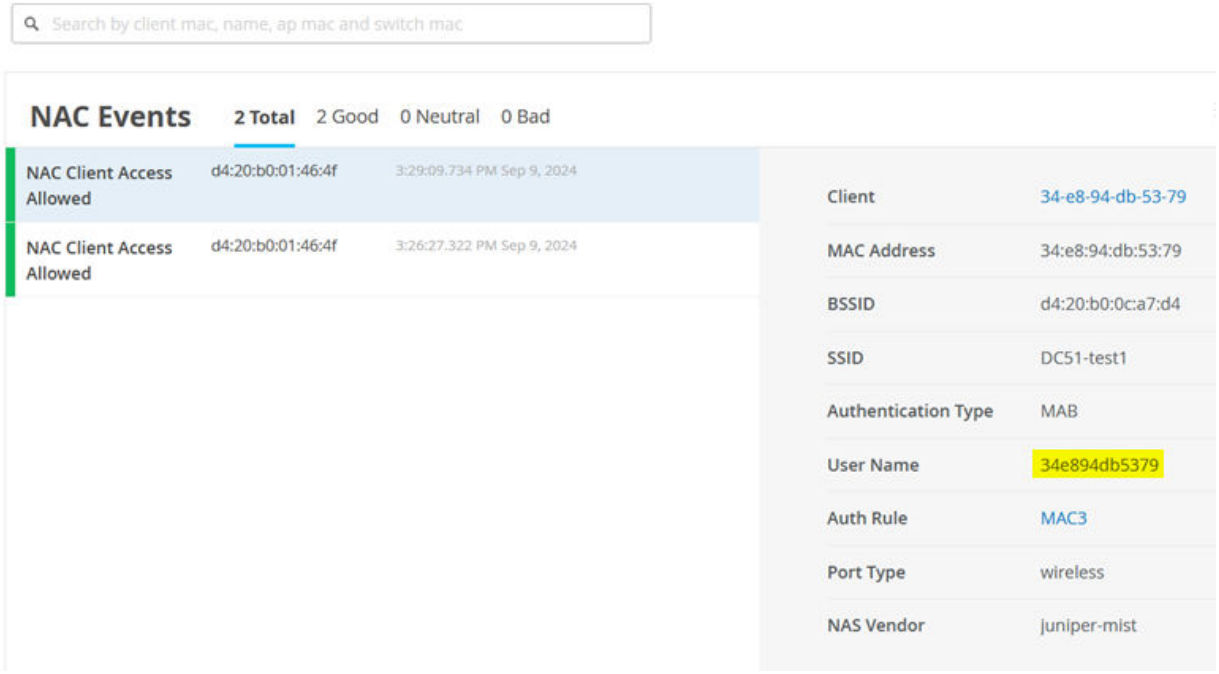
Through a second shell, obtain a DHCP lease and check the AP assignment:

```
root@desktop3:~# dhclient wlx34e894db5379
root@desktop3:~# ip r
default via 10.88.88.1 dev wlx34e894db5379
10.88.88.0/24 dev wlx34e894db5379 proto kernel scope link src 10.88.88.10
root@desktop3:~# iwconfig
ens3      no wireless extensions.
wlx34e894db5379  IEEE 802.11AC  ESSID:"DC51-test1"  Nickname:"WIFI@RTL8821AU"
          Mode:Managed  Frequency:5.26 GHz  Access Point: D4:20:B0:0C:A7:D4
          Bit Rate:200 Mb/s   Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:****-****-****-****-****-****-****-****   Security mode:open
          Power Management:off
          Link Quality=58/100  Signal level=75/100  Noise level=0/100
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
lo        no wireless extensions.
```

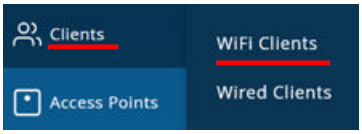
Return to **Organization > Auth Policies** and you should see the **Hit Count** for the rule incremented like in the figure shown below:



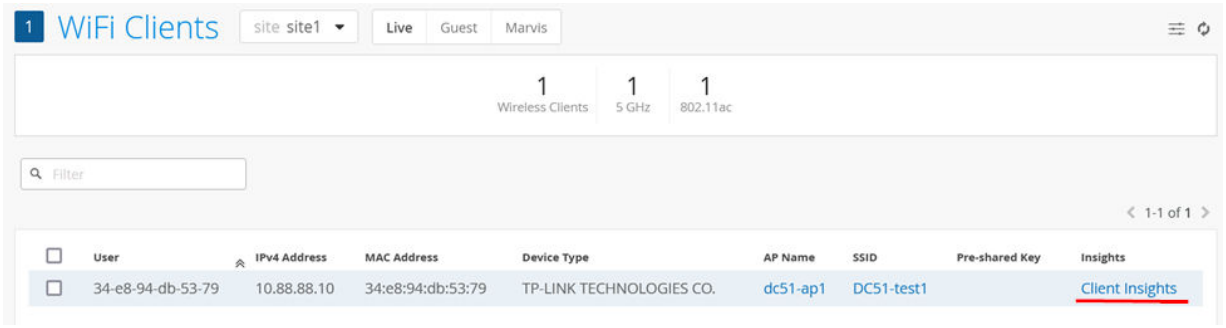
Click on this link and you should see information like the figure shown below:



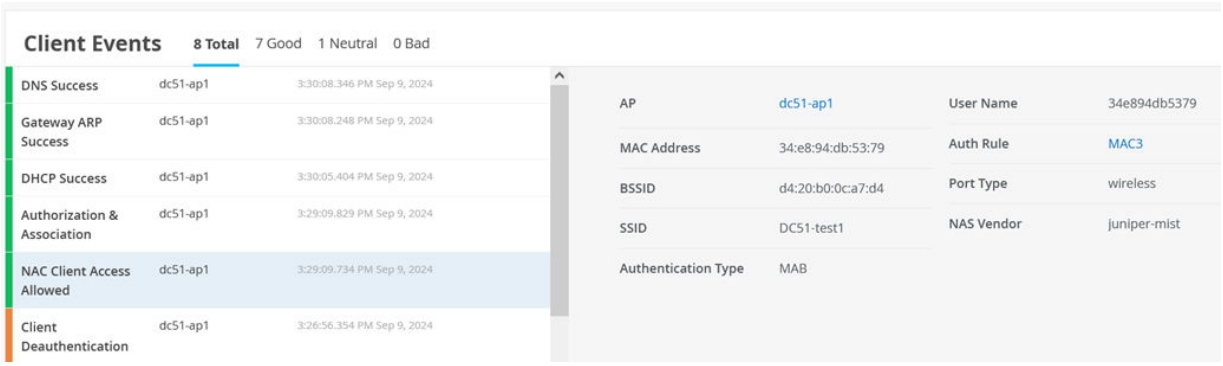
Another way to review information about the wireless client is to go to **Clients > WiFi Clients.**



You should see your client, and where it is attached. Click on **Client Insights.**



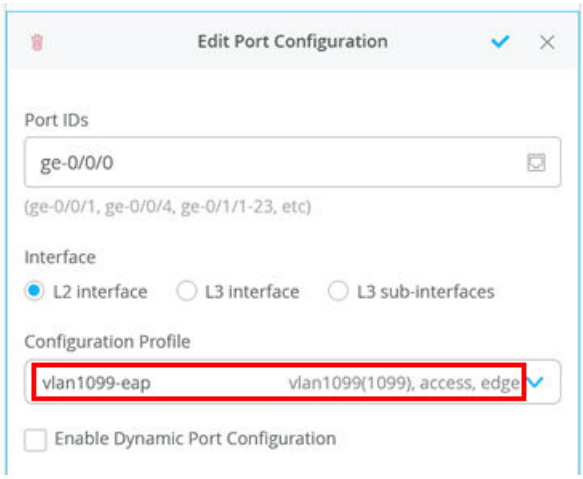
In the **Client Events** section, you will see information about the MAC address-based authentication process.



EAP-TLS Authentication of a Wired Client

To test EAP-TLS wired client-based authentication of a wired client, execute the following steps one by one.

First, we need to change the port on the access switch where the wired client is attached to use the profile for 802.1X that we defined in the switch template in Figure 26 on page 84. Change the configuration profile to `vlan1099-eap`:



NOTE: After this is applied, your wired clients will not be able to communicate further with the network as we have not authenticated them yet.

(Optional) Remote Shell to the switch to review the configurations applied for RadSec, the certificate, and the port.

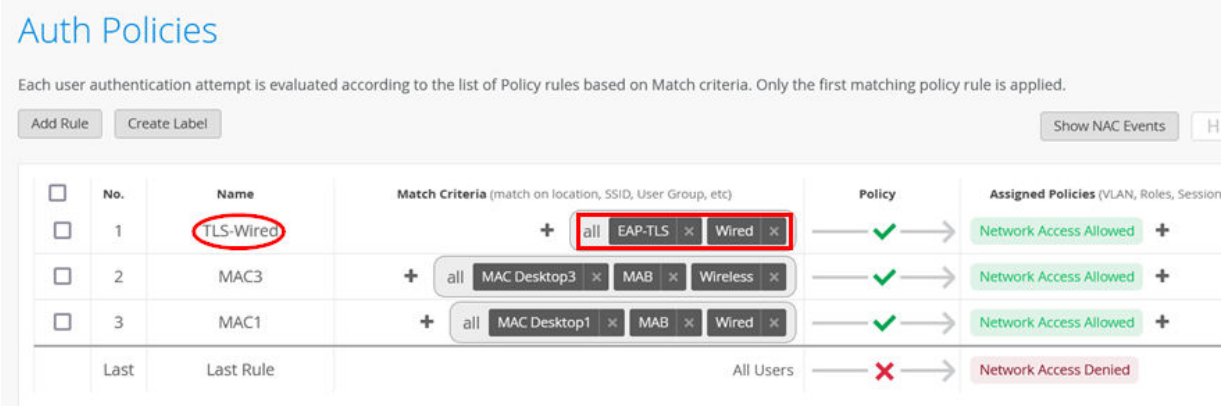
```
mist@switch1> show configuration | display set | match dot1x
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37700
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set protocols dot1x authenticator authentication-profile-name dot1x
set protocols dot1x authenticator interface vlan1099-eap
mist@switch1> show configuration | display set | match vlan1099-eap | match interfaces
set groups vlan1099-eap interfaces <*> unit 0 family ethernet-switching vlan members vlan1099
set interfaces interface-range vlan1099-eap member ge-0/0/0
set interfaces interface-range vlan1099-eap apply-groups vlan1099-eap
mist@switch1> show configuration | display set | match access
set groups top access radius-server 3.33.153.159 secret "$9$gnaDk.mTn6AP5nCu0hc"
set groups top access radius-server 3.33.153.159 timeout 5
set groups top access radius-server 3.33.153.159 radsec-destination 895
set groups top access radius-server 15.197.139.214 secret "$9$gnaDk.mTn6AP5nCu0hc"
set groups top access radius-server 15.197.139.214 timeout 5
set groups top access radius-server 15.197.139.214 radsec-destination 896
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37700
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set groups top access radsec destination 895 address 3.33.153.159
set groups top access radsec destination 895 port 2083
set groups top access radsec destination 895 tls-certificate mist-nac-device-cert
set groups top access radsec destination 895 tls-force-ciphers low
set groups top access radsec destination 895 tls-min-version v1.2
```

```
set groups top access radsec destination 895 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 895 tls-timeout 30
set groups top access radsec destination 896 address 15.197.139.214
set groups top access radsec destination 896 port 2083
set groups top access radsec destination 896 tls-certificate mist-nac-device-cert
set groups top access radsec destination 896 tls-force-ciphers low
set groups top access radsec destination 896 tls-min-version v1.2
set groups top access radsec destination 896 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 896 tls-timeout 30
mist@switch1> show security pki ca-certificate
LSYS: root-logical-system
  CA profile: mist-vpn-ca
Certificate identifier: mist-vpn-ca
  Issued to: cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae, Issued by: C = US, O = Mist, OU = OrgCA, CN =
cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-28-2023 09:37 UTC
    Not after: 08-25-2033 09:37 UTC
  Public key algorithm: rsaEncryption(4096 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show security pki local-certificate
LSYS: root-logical-system
Certificate identifier: mist-nac-device-cert
  Issued to: 00cc34f37400, Issued by: C = US, O = Mist, OU = OrgCA, CN = cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-23-2024 10:11 UTC
    Not after: 08-23-2025 10:11 UTC
  Public key algorithm: rsaEncryption(2048 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show system connections | match 2083
tcp4       0      0  10.33.33.19.59751
15.197.139.214.2083                    ESTABLISHED
tcp4       0      0  10.33.33.19.65050
3.33.153.159.2083                      ESTABLISHED
mist@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/0.0    Authenticator  Connecting
mist@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
```

```
Role: Authenticator
Administrative state: Auto
Supplicant mode: Single
Number of retries: 3
Quiet period: 60 seconds
Transmit period: 30 seconds
Mac Radius: Disabled
Mac Radius Restrict: Disabled
Reauthentication: Enabled
Reauthentication interval: 3600 seconds
Supplicant timeout: 30 seconds
Server timeout: 30 seconds
Maximum EAPOL requests: 2
Guest VLAN member: not configured
Number of connected supplicants: 0
```

We do not need to specify a label as we only use the authentication type and the port location for identification of the client in this example. So, go to **Organization > Auth Policies** and create the following rule:

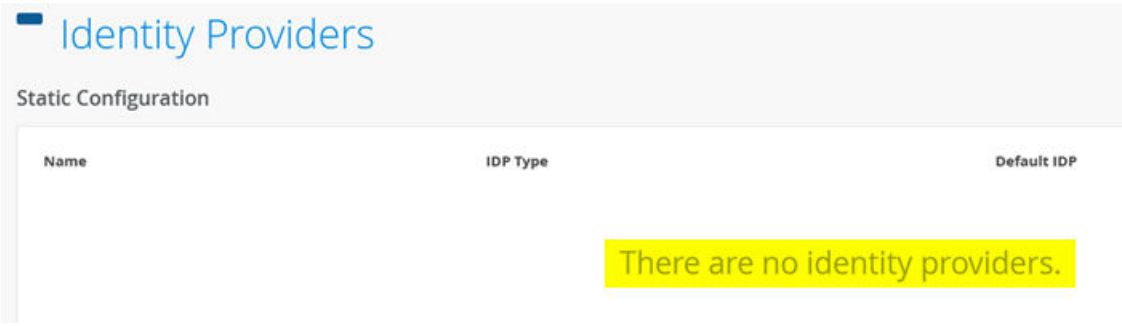
- Position=`1`
- Name=`TLS-Wired`
- Match Criteria=`EAP-TLS` and `Wired`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



If you have not done it already, you must perform the enterprise PKI integration and let the Juniper Mist authentication cloud learn the root CA and install a TLS server certificate/key for the RADIUS server.

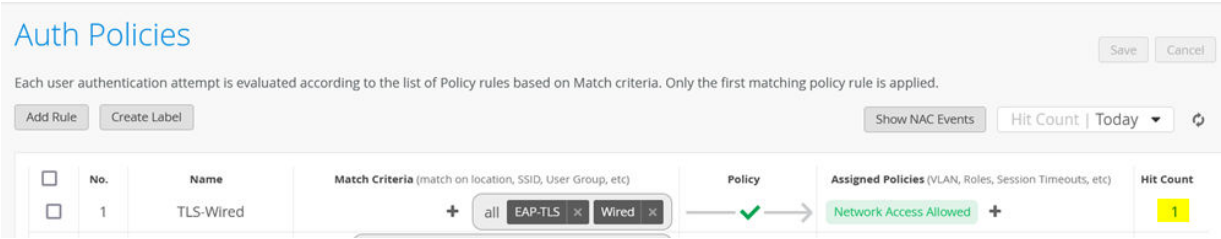
Refer to the examples in the section "Juniper Mist Authentication Cloud Certificate Installation" on page 114.

For this test, ensure that you have not configured any IdP yet since we rely only on the validity of the certificates on both the RADIUS and supplicant sides.

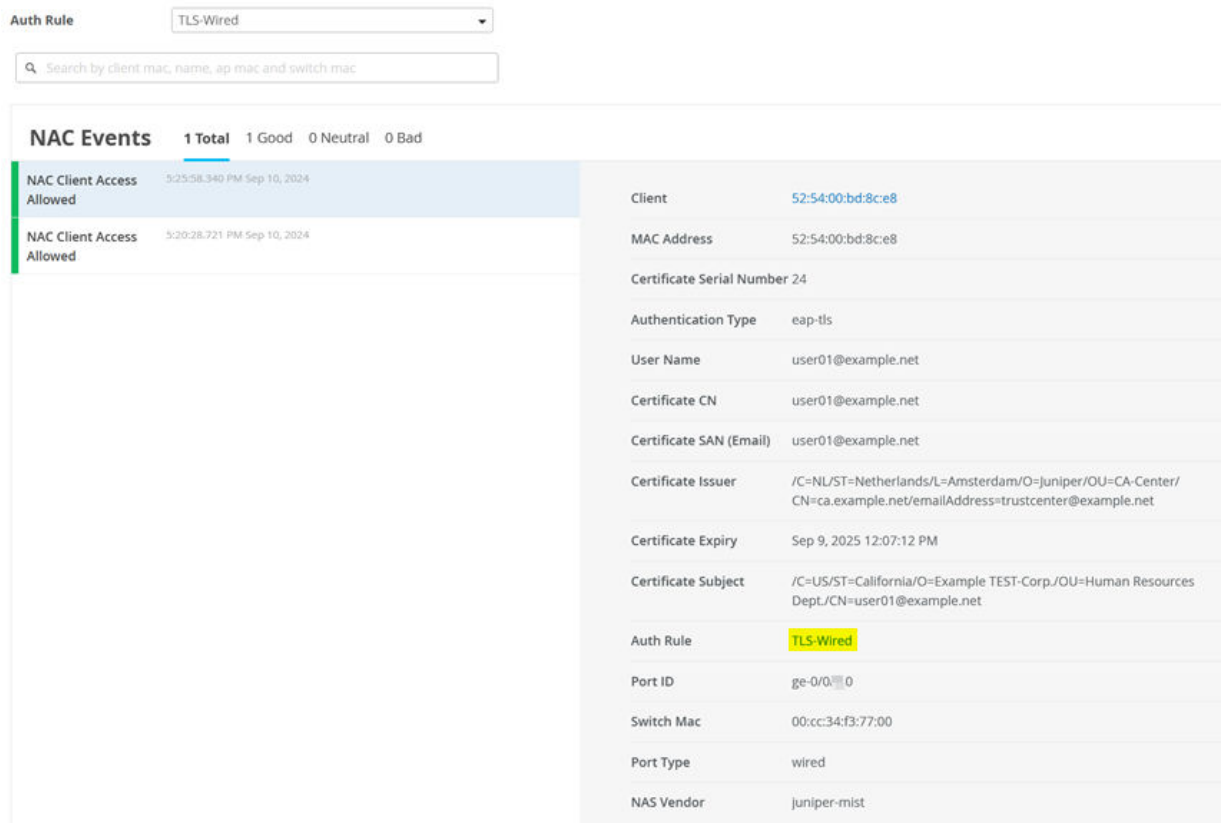


Next, perform an EAP-TLS authentication with a wired supplicant relevant to your client operating system. We have shared examples of configurations for Windows and Linux clients in the section "Configure Client Supplicants with Certificates and Necessary EAP Methods" on page 119.

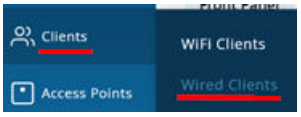
Upon successful completion of the EAP-TLS authentication, you should see the **Hit Count** incremented similar to the figure shown below:



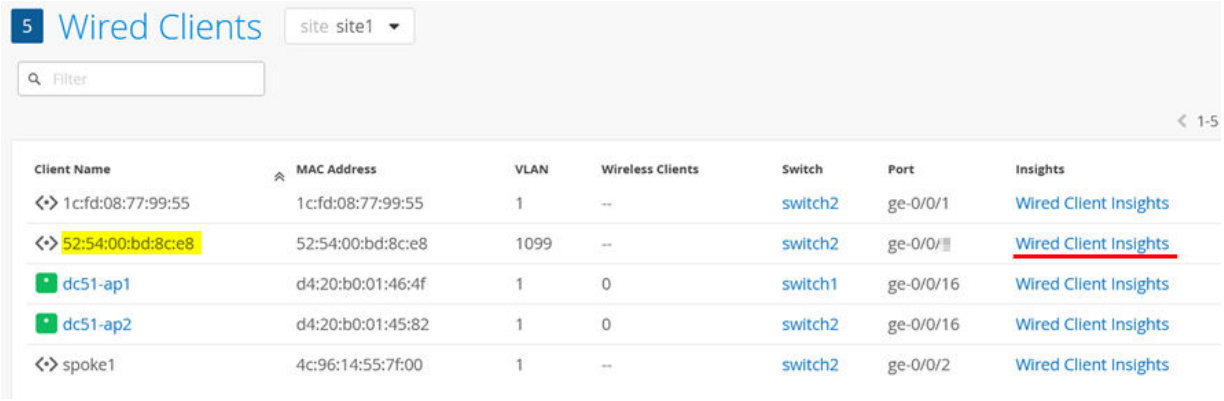
You can see the information about your client:



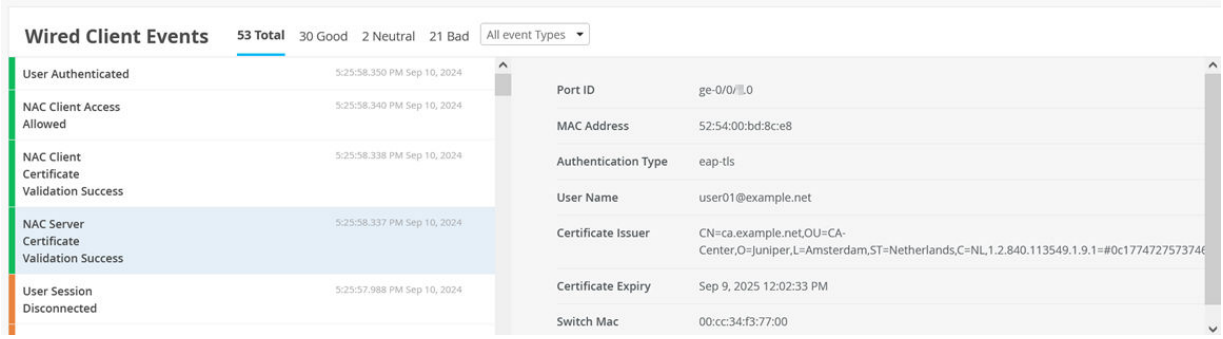
You can also go to **Clients > Wired Clients**.



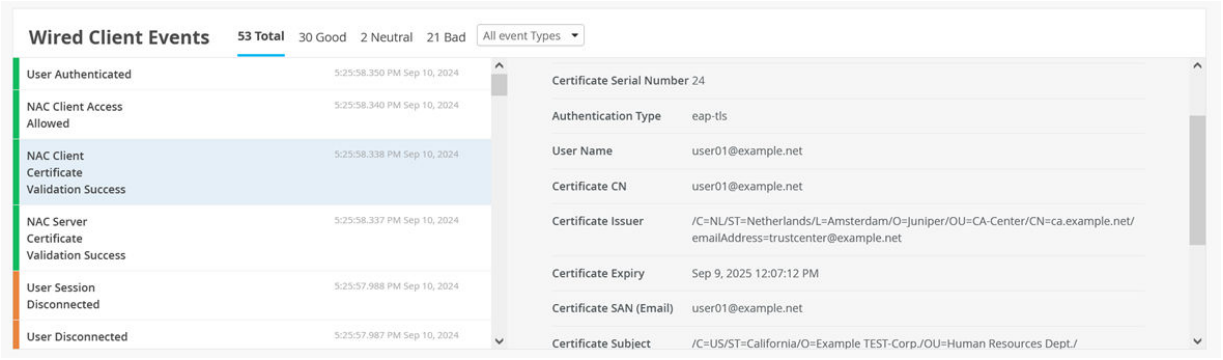
Then, find your client and select **Wired Client Insights.**



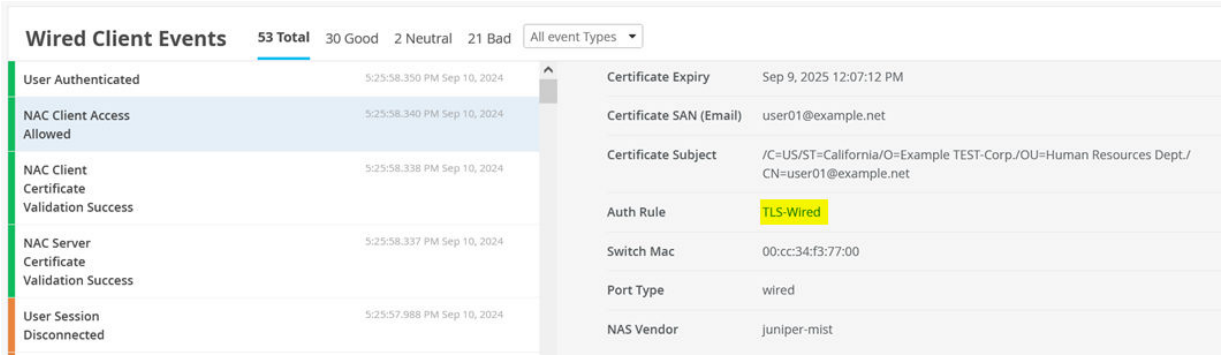
Then, you can inspect the **Wired Client Events** for **NAC Server Certificate Validation Success** events.



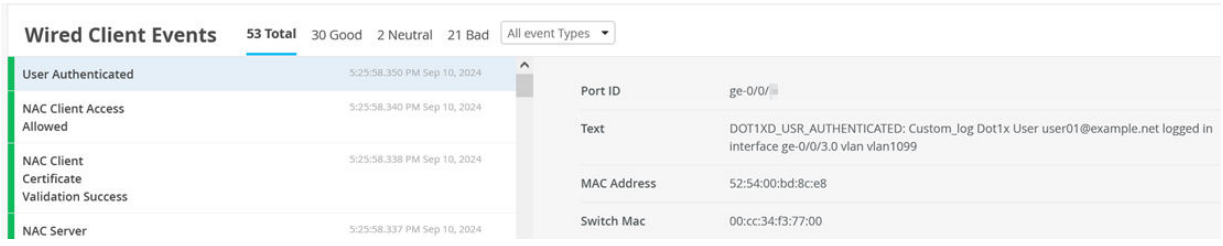
Also see **NAC Client Certificate Validation Success** events.



And **NAC Client Access Allowed** events.



And **User Authenticated** events.



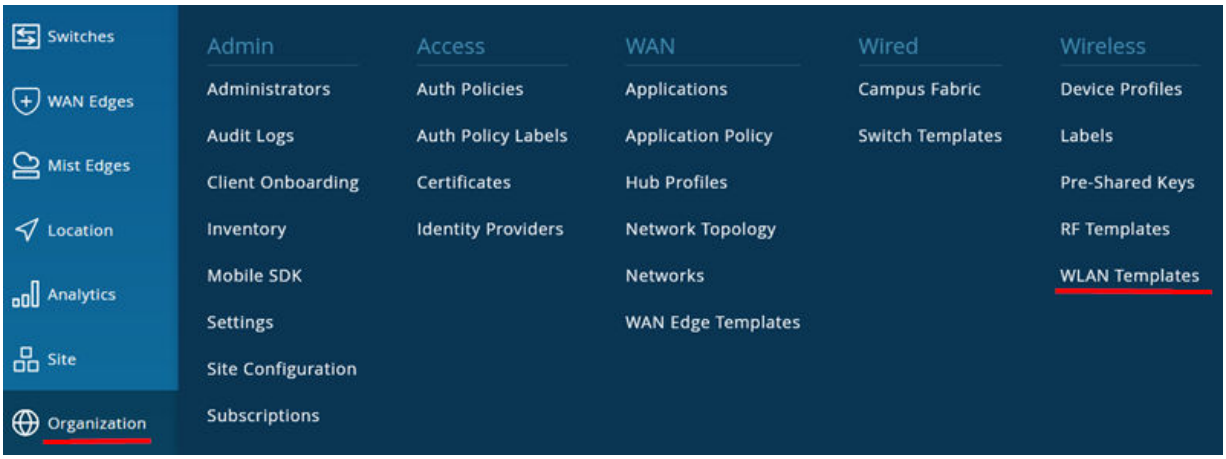
(Optional) Remote Shell to the switch and check the client authentication status using commands like those shown below:

```
root@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/0.0    Authenticator  Authenticated   52:54:00:BD:8C:E8    user01@example.net
root@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
  Supplicant timeout: 30 seconds
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 1
    Supplicant: user01@example.net, 52:54:00:BD:8C:E8
      Operational state: Authenticated
      Backend Authentication state: Idle
      Authentication method: Radius
      Authenticated VLAN: vlan1099
      Session Reauth interval: 3600 seconds
      Reauthentication due in 3287 seconds
      Session Accounting Interim  Interval: 36000 seconds
      Accounting Update  due in 35687 seconds
      Eapol-Block: Not In Effect
      Domain: Data
```

EAP-TLS Authentication of a Wireless Client

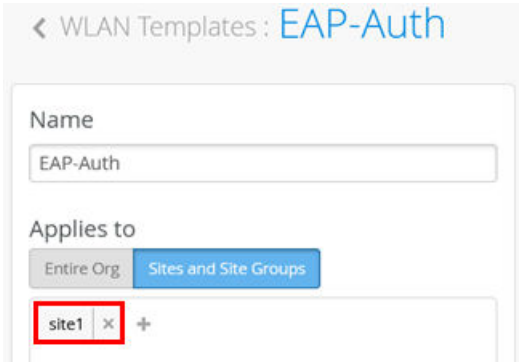
In this example, we use EAP-TLS for 802.1X-based authentication of wireless clients.

An SSID for wireless can be configured in several ways. In our example, we use a WLAN template by first navigating to **Organization > WLAN Templates**.

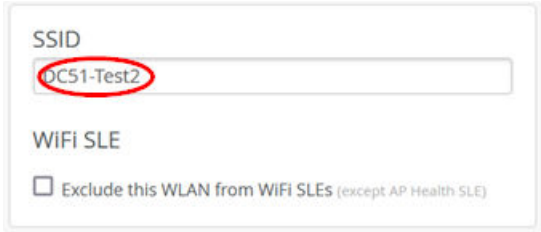


Create a new template with the following settings:

- Name=`EAP-Auth`
- Applies to:
  - Sites and Site Groups=`site1`



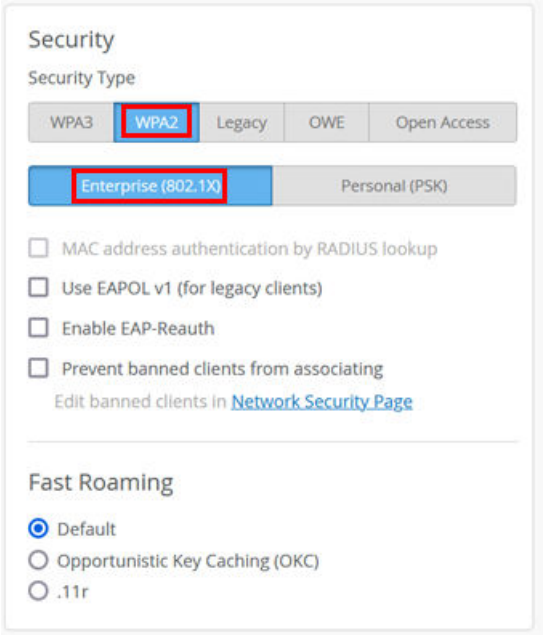
Create an SSID like the figure shown below:



Under **Security**, configure the following settings:

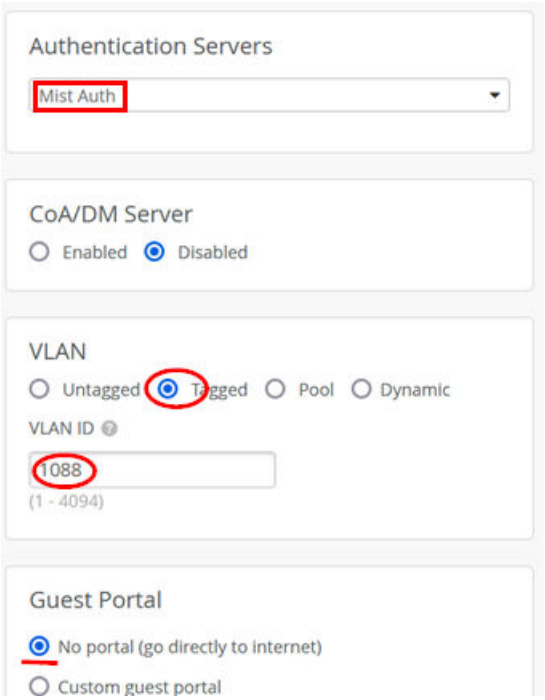
- Security Type=`WPA2` and

```
Enterprise
(802.1X)
```

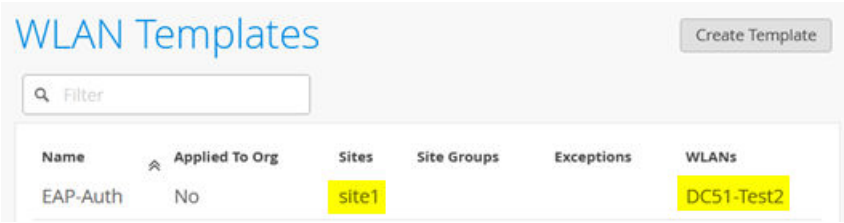


Then, configure the following settings:

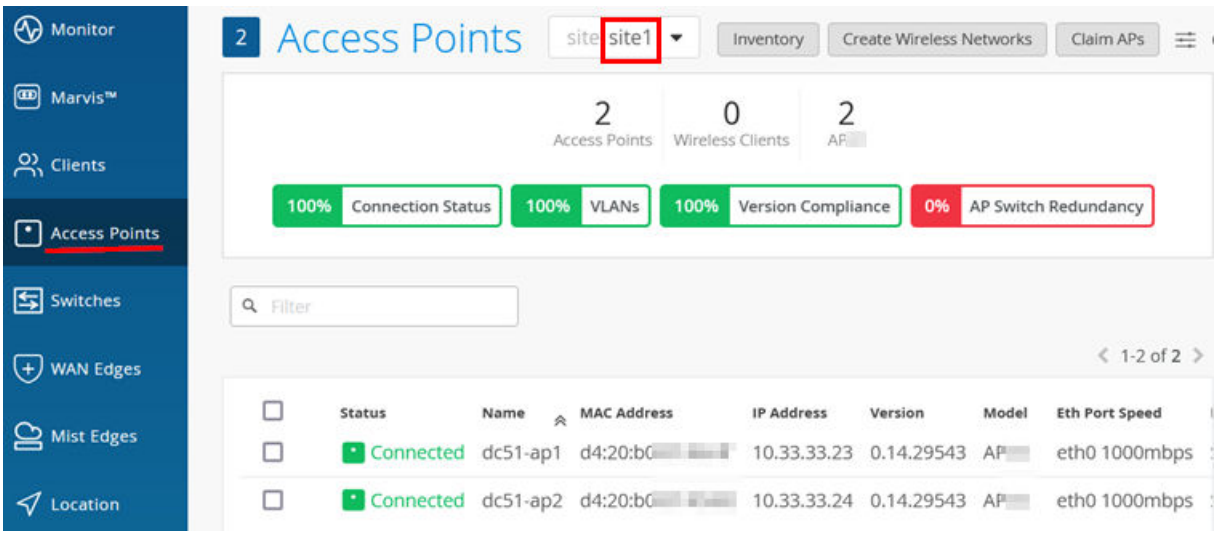
- Authentication Servers=`Mist Auth`
- VLAN=`Tagged`
- VLAN ID=`1088`



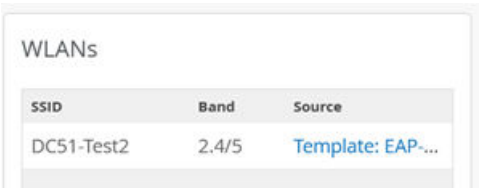
After saving the template, you should see the following configuration:



Next, go **Access Points** and select your site to review the APs. Select one AP.

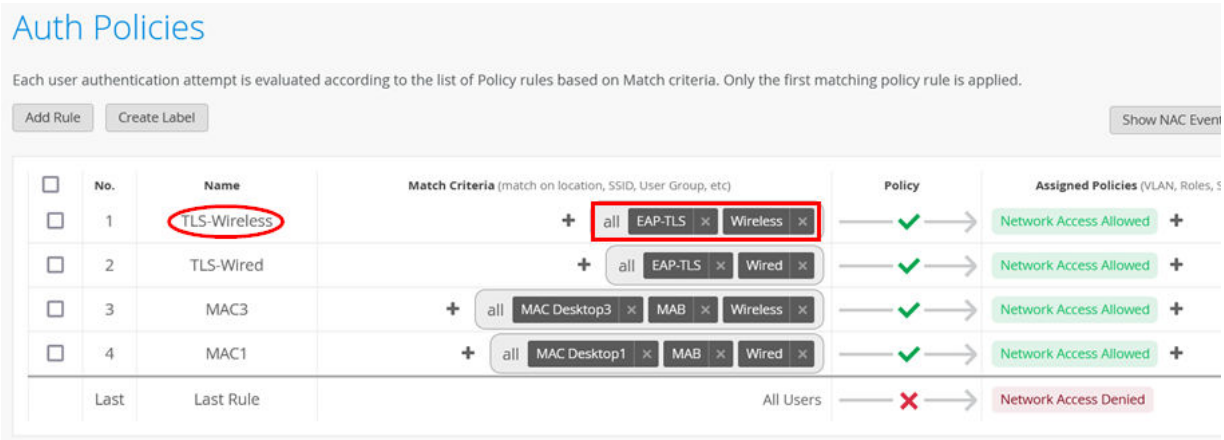


Review the AP configuration applied, making sure the SSID from the template appears in the WLANs tab:



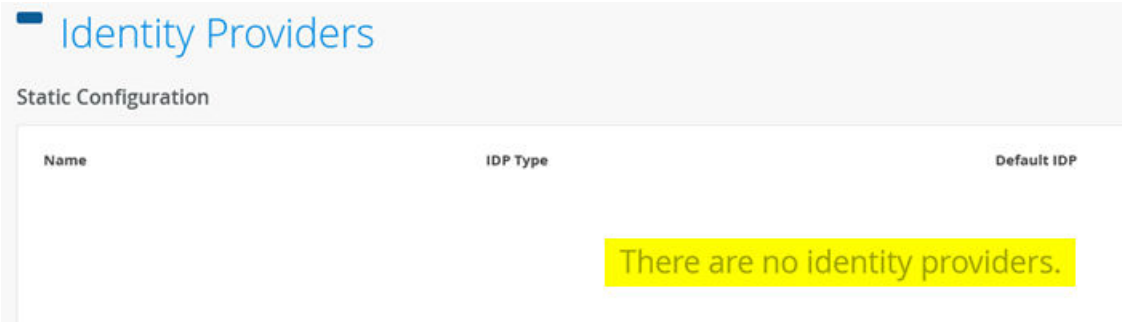
We do not need to specify a label as we only use the authentication type and the port location for identification of the client in this example. So, go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`TLS-Wireless`
- Match Criteria=`EAP-TLS` and `Wireless`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



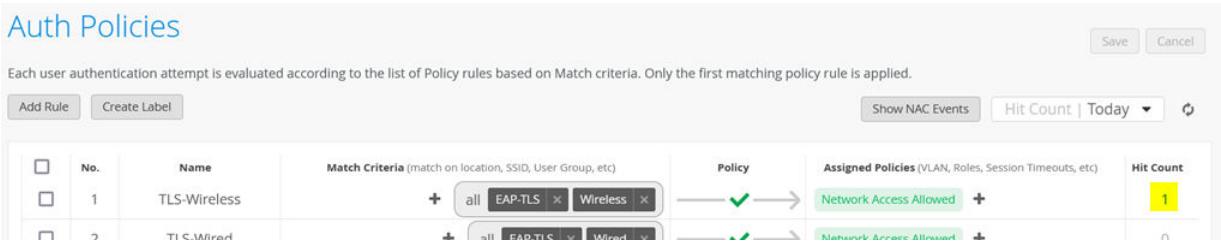
If you have not done it already you must perform the enterprise PKI integration and let the Juniper Mist authentication cloud learn the root CA and install a TLS server certificate/key for the RADIUS server. Refer to the examples in the section "Mist Authentication Cloud Certificate Installation" on page 114.

For this test, ensure that you have not configured any IdP yet since we only rely on the validity of the certificates on the RADIUS and supplicant sides.

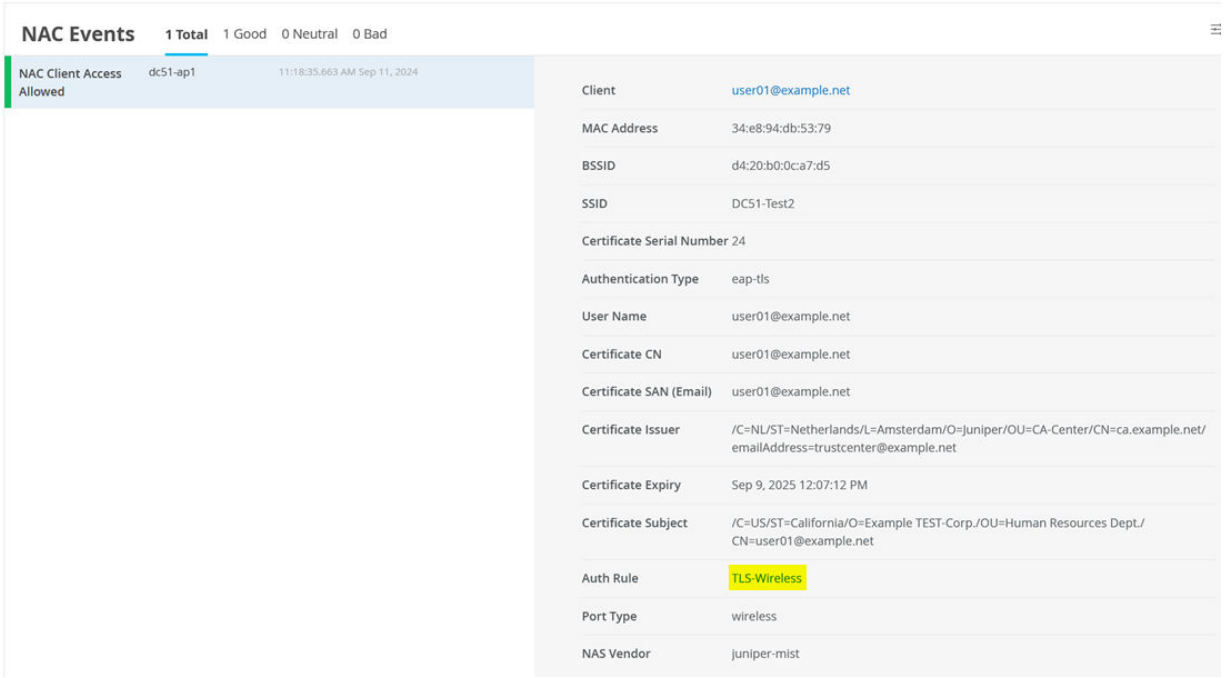


Next, perform EAP-TLS authentication with a wireless supplicant relevant to your client operating system. We have shared examples of configurations for Windows and Linux clients in the section "Configure Client Supplicants with Certificates and Necessary EAP Methods" on page 119.

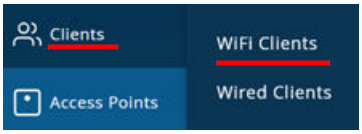
Upon successful completion of the EAP-TLS authentication, you should see the **Hit Count** increment like the figure shown below:



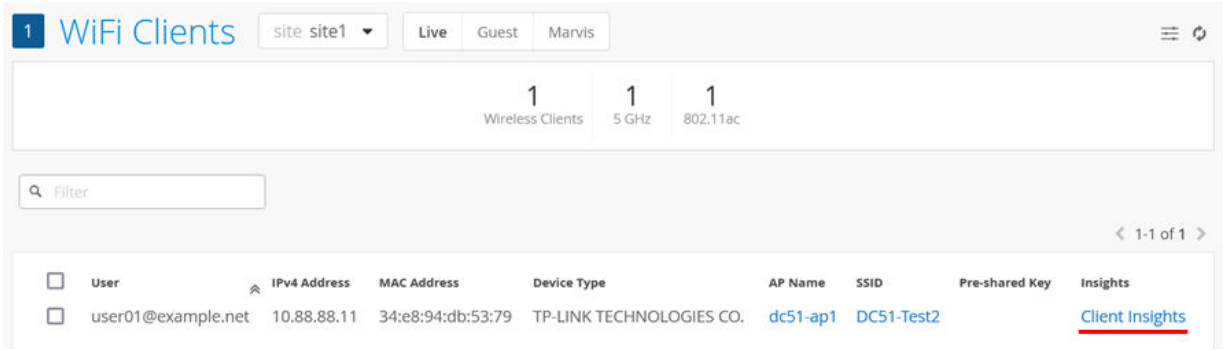
When you click this link, you will see information about the authentication event:



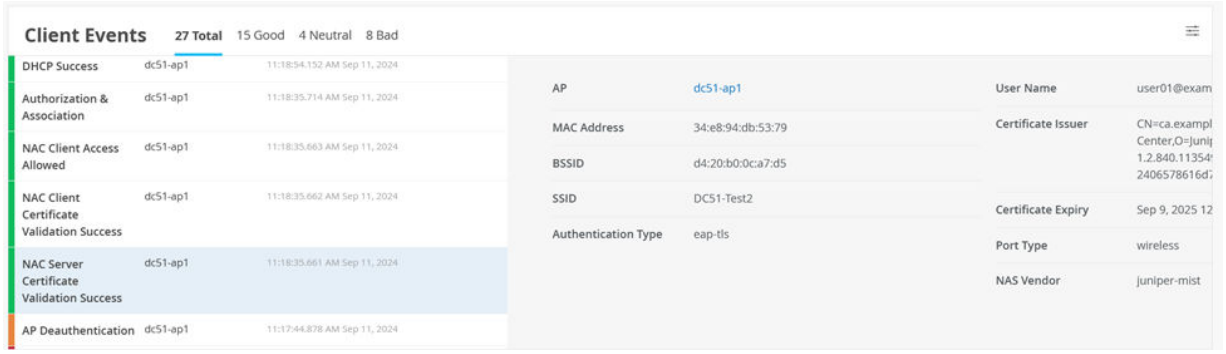
Another way to review information about the wireless client is to go to **Clients > WiFi Clients**.



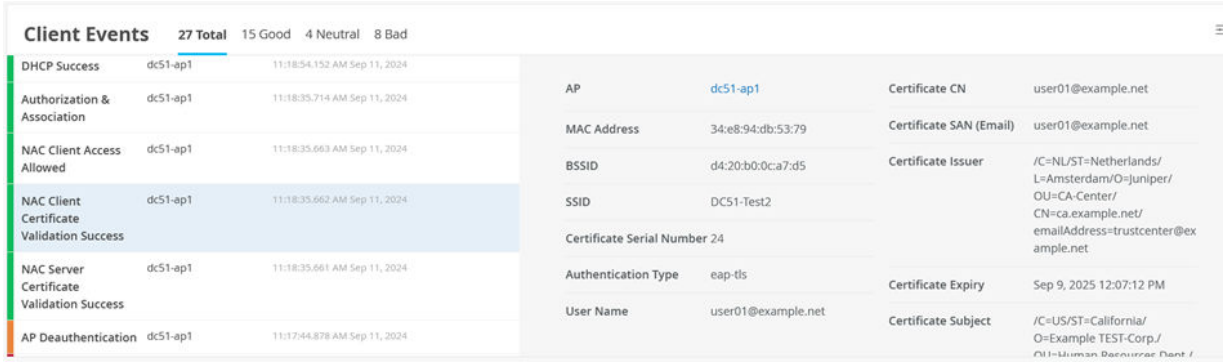
You should see your client and where it is attached. Click on **Client Insights.**



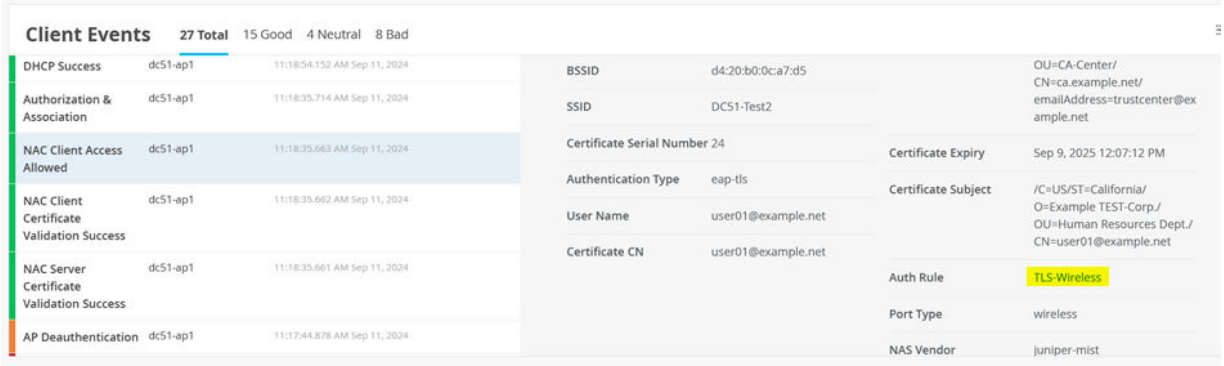
In the **Client Events** section, you will see information about the EAP-TLS-based authentication process. For instance, the **NAC Server Certificate Validation Success** event.



Also see **NAC Client Certificate Validation Success** events.



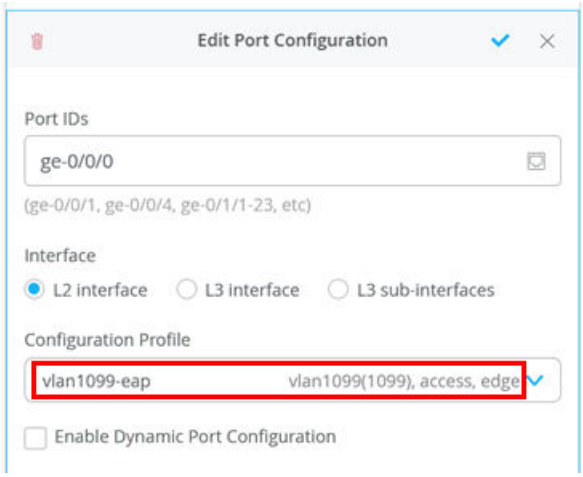
And **NAC Client Access Allowed** events.



EAP-TTLS Authentication of a Wired Client

To test EAP-TTLS client-based authentication of a wired client, execute the following steps one by one.

First, we need to change the port on the access switch where the wired client is attached to use the profile for 802.1X that we defined in the switch template in Figure 26 on page 84. Change the configuration profile to `vlan1099-eap`.



NOTE: After this is applied, your wired clients will not be able to communicate further with the network as we have not authenticated them yet.

(Optional) Remote Shell to the switch to review the configuration applied for RadSec, the certificate, and the port.

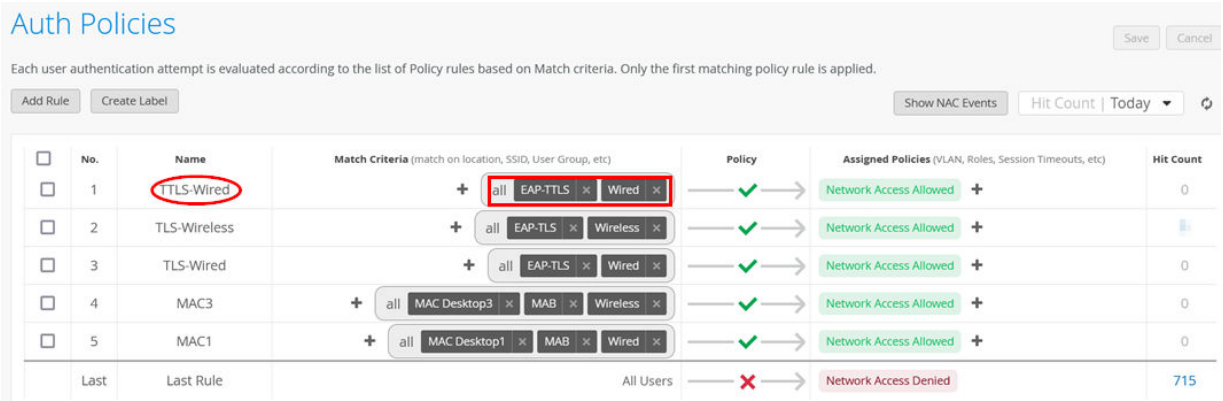
```
mist@switch1> show configuration | display set | match dot1x
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37700
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set protocols dot1x authenticator authentication-profile-name dot1x
set protocols dot1x authenticator interface vlan1099-eap
mist@switch1> show configuration | display set | match vlan1099-eap | match interfaces
set groups vlan1099-eap interfaces <*> unit 0 family ethernet-switching vlan members vlan1099
set interfaces interface-range vlan1099-eap member ge-0/0/0
set interfaces interface-range vlan1099-eap apply-groups vlan1099-eap
mist@switch1> show configuration | display set | match access
set groups top access radius-server 3.33.153.159 secret "$9$gnaDk.mTn6AP5nCu0hc"
set groups top access radius-server 3.33.153.159 timeout 5
set groups top access radius-server 3.33.153.159 radsec-destination 895
set groups top access radius-server 15.197.139.214 secret "$9$gnaDk.mTn6AP5nCu0hc"
set groups top access radius-server 15.197.139.214 timeout 5
set groups top access radius-server 15.197.139.214 radsec-destination 896
set groups top access profile dot1x accounting-order radius
set groups top access profile dot1x authentication-order radius
set groups top access profile dot1x radius authentication-server 3.33.153.159
set groups top access profile dot1x radius authentication-server 15.197.139.214
set groups top access profile dot1x radius accounting-server 3.33.153.159
set groups top access profile dot1x radius accounting-server 15.197.139.214
set groups top access profile dot1x radius options nas-identifier
6ce2ec31-4db2-4d56-8aae-4047380273cb00cc34f37700
set groups top access profile dot1x accounting order radius
set groups top access profile dot1x accounting update-interval 600
set groups top access radsec destination 895 address 3.33.153.159
set groups top access radsec destination 895 port 2083
set groups top access radsec destination 895 tls-certificate mist-nac-device-cert
set groups top access radsec destination 895 tls-force-ciphers low
set groups top access radsec destination 895 tls-min-version v1.2
```

```
set groups top access radsec destination 895 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 895 tls-timeout 30
set groups top access radsec destination 896 address 15.197.139.214
set groups top access radsec destination 896 port 2083
set groups top access radsec destination 896 tls-certificate mist-nac-device-cert
set groups top access radsec destination 896 tls-force-ciphers low
set groups top access radsec destination 896 tls-min-version v1.2
set groups top access radsec destination 896 tls-peer-name aws-production.cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
set groups top access radsec destination 896 tls-timeout 30
mist@switch1> show security pki ca-certificate
LSYS: root-logical-system
  CA profile: mist-vpn-ca
Certificate identifier: mist-vpn-ca
  Issued to: cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae, Issued by: C = US, O = Mist, OU = OrgCA, CN =
cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-28-2023 09:37 UTC
    Not after: 08-25-2033 09:37 UTC
  Public key algorithm: rsaEncryption(4096 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show security pki local-certificate
LSYS: root-logical-system
Certificate identifier: mist-nac-device-cert
  Issued to: 00cc34f37400, Issued by: C = US, O = Mist, OU = OrgCA, CN = cdd0e7d1-
e1f2-4280-86cd-0327e6ce88ae
  Validity:
    Not before: 08-23-2024 10:11 UTC
    Not after: 08-23-2025 10:11 UTC
  Public key algorithm: rsaEncryption(2048 bits)
  Keypair Location: Keypair generated locally
mist@switch1> show system connections | match 2083
tcp4       0      0  10.33.33.19.59751
15.197.139.214.2083                    ESTABLISHED
tcp4       0      0  10.33.33.19.65050
3.33.153.159.2083                      ESTABLISHED
mist@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/0.0    Authenticator  Connecting
mist@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
```

```
Role: Authenticator
Administrative state: Auto
Supplicant mode: Single
Number of retries: 3
Quiet period: 60 seconds
Transmit period: 30 seconds
Mac Radius: Disabled
Mac Radius Restrict: Disabled
Reauthentication: Enabled
Reauthentication interval: 3600 seconds
Supplicant timeout: 30 seconds
Server timeout: 30 seconds
Maximum EAPOL requests: 2
Guest VLAN member: not configured
Number of connected supplicants: 0
```

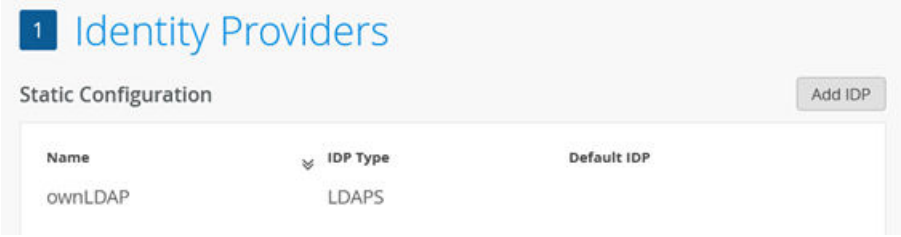
We do not need to specify a label as we only use the authentication type and the port location for identification of the client in this example. So, go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`TTLS-Wired`
- Match Criteria=`EAP-TTLS` and `Wired`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



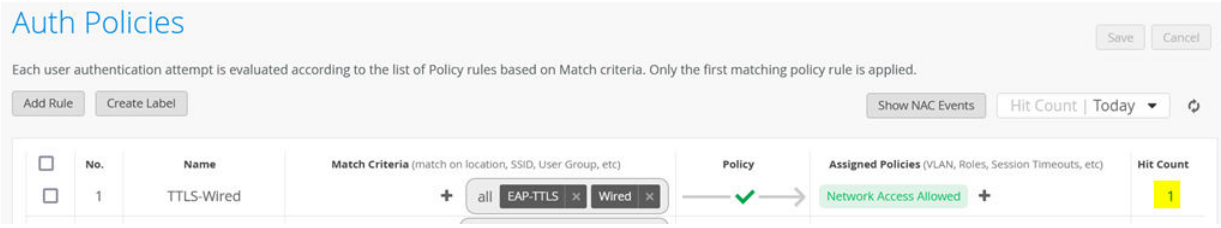
If you have not done it already you must perform the enterprise PKI integration and let the Juniper Mist authentication cloud learn the root CA and install a TLS server certificate/key for the RADIUS server. Refer to the examples in the section "Juniper Mist Authentication Cloud Certificate Installation" on page 114.

For this test, it is mandatory to have at least one IdP specified since we need to perform a credential check. Hence, the RADIUS server needs to be able to contact a credential database. In our example, we leverage a simplistic LDAP repository. In a production-grade environment, you would probably use Azure or Okta instead. Remember that we have provided examples for those integrations in the section "Configuration Examples of Public Identity Provider Database Integration" on page 154.

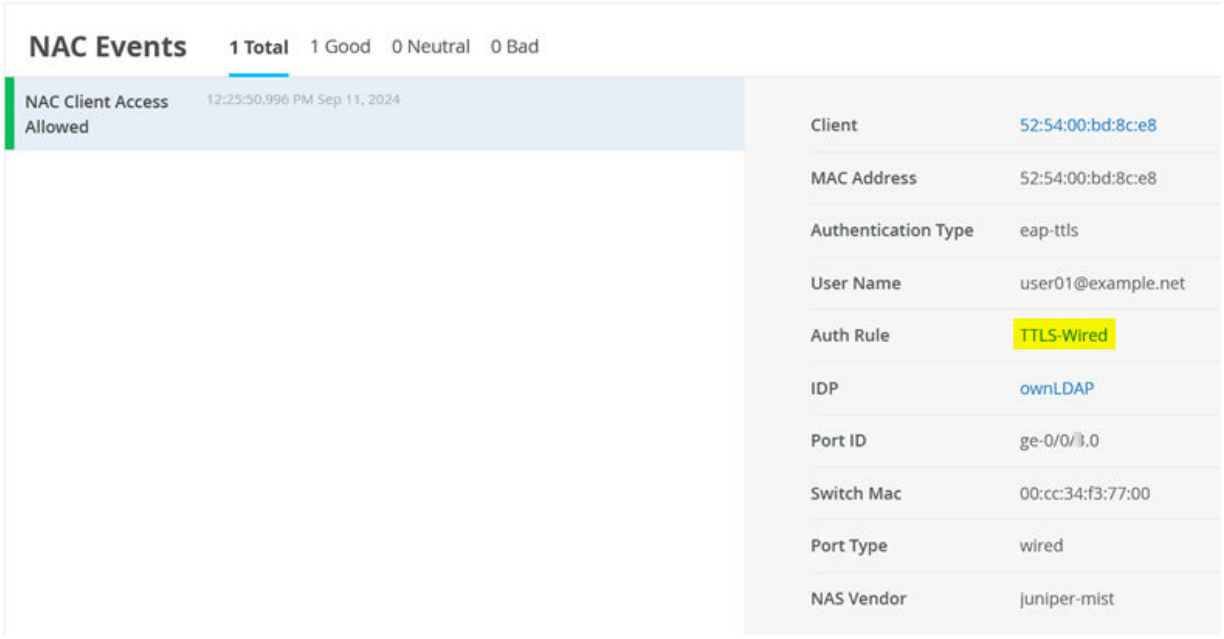


Next, you need to perform EAP-TTLS authentication with a wired supplicant relevant to your client operating system. We have shared examples of such configurations for Windows and Linux clients in the section "Configure Client Supplicants with Certificates and Necessary EAP Methods" on page 119.

Upon successful completion of the EAP-TTLS authentication, you should see the **Hit Count** incremented like in the figure shown below:

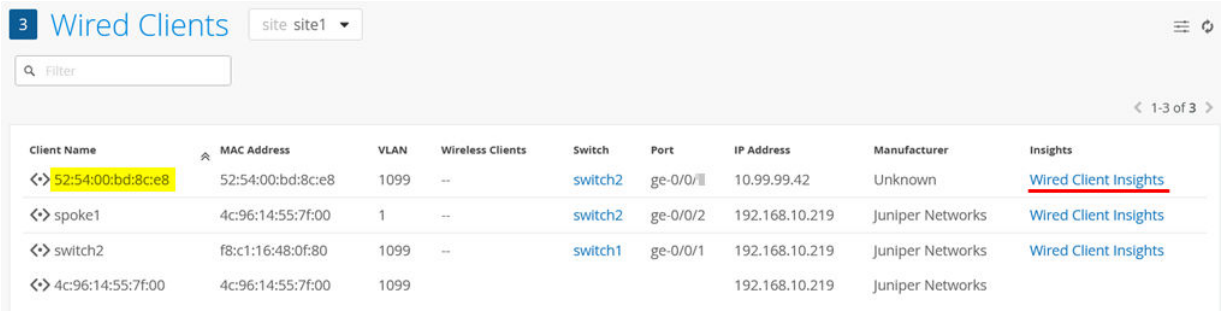


You can see the information about your client:

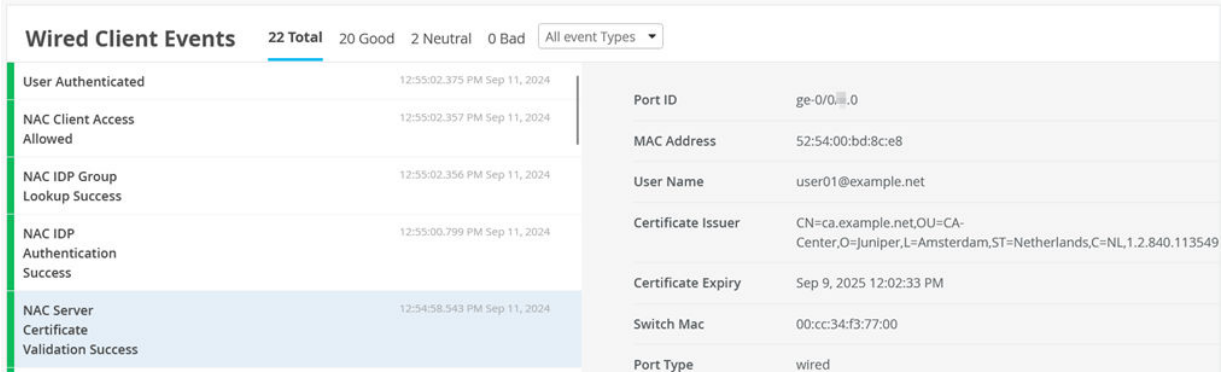


You can also go to **Clients > Wired Clients**.

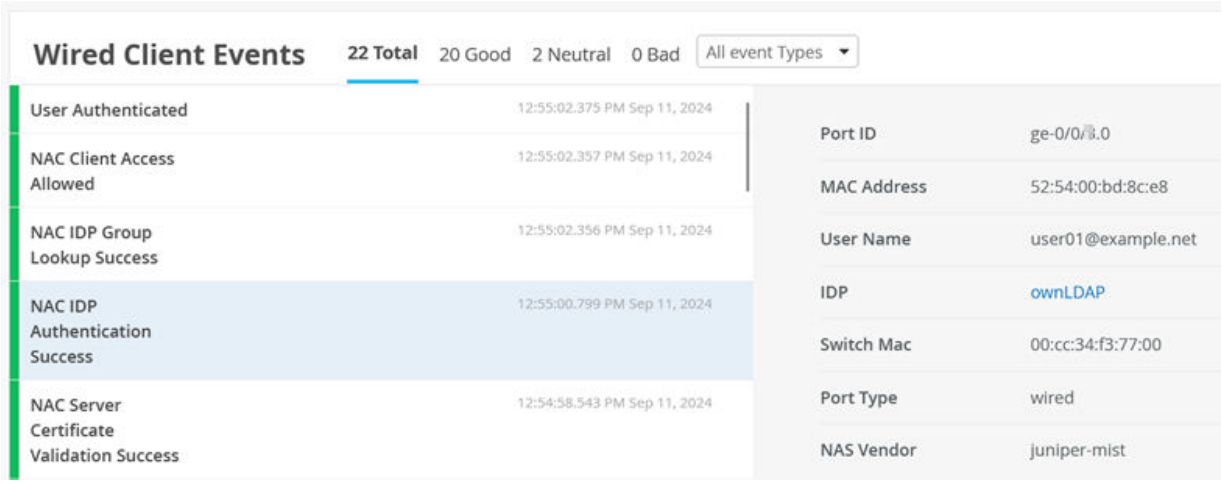
Then, look for your client and select **Wired Client Insights**.



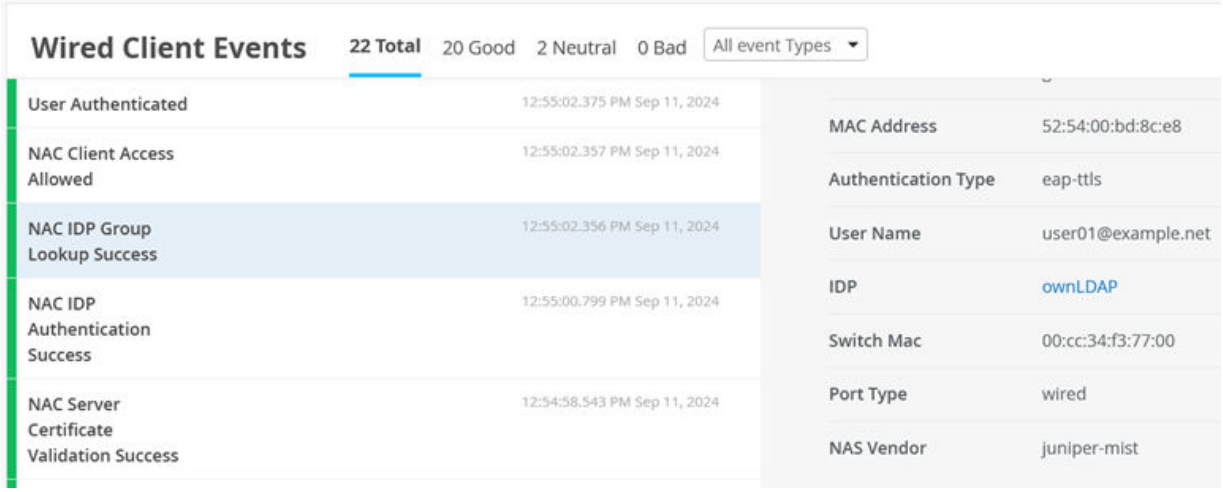
Then, you can inspect the **Wired Client Events** for **NAC Server Certificate Validation Success** events.



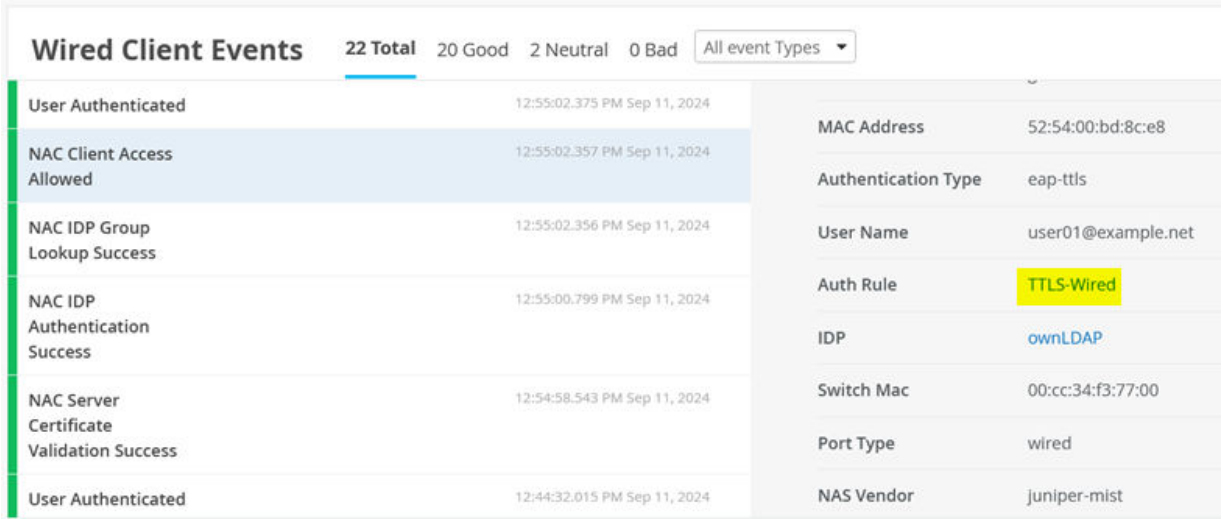
Also see **NAC IDP Authentication Success** events.



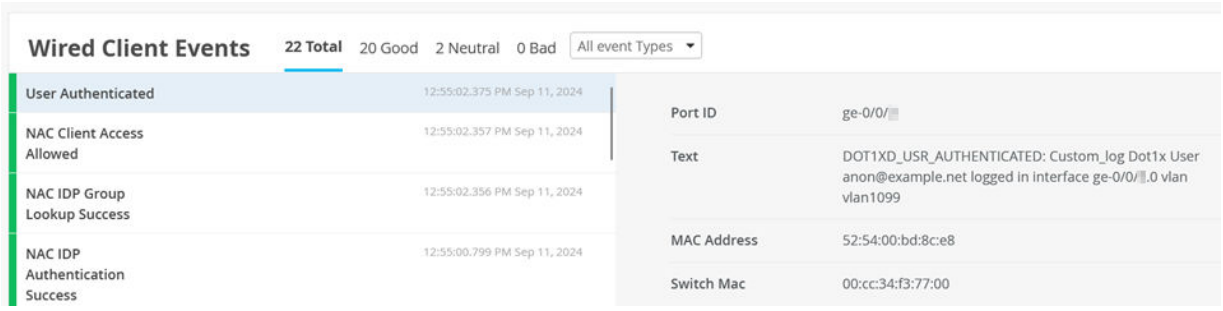
And **NAC IDP Group Lookup Success** events.



And **NAC Client Access Allowed** events.



And **User Authenticated** events.



(Optional) Remote Shell to the switch and check the client authentication status using commands like those shown below:

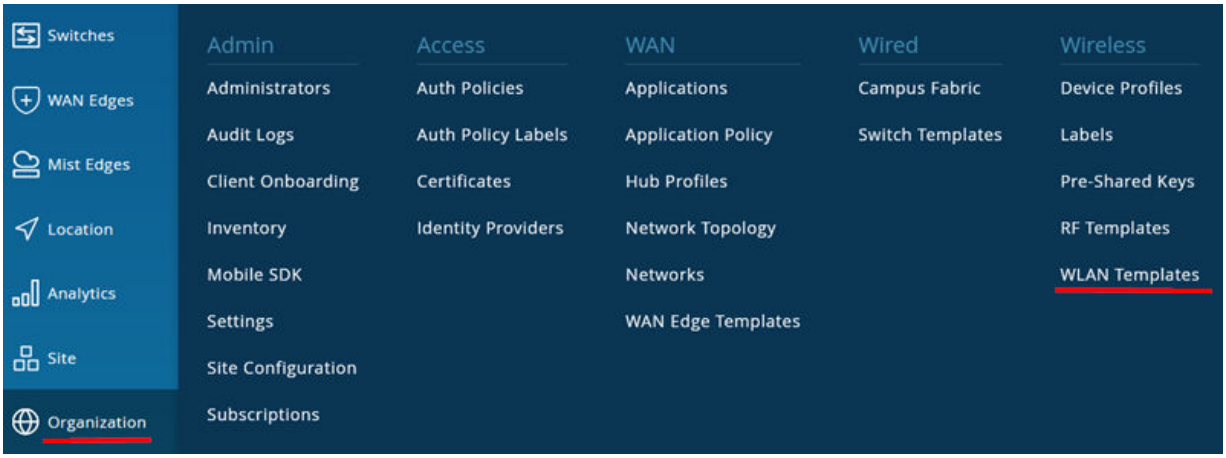
```
root@switch1> show dot1x interface ge-0/0/0
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/0.0    Authenticator  Authenticated   52:54:00:BD:8C:E8    user01@example.net
root@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
```

```
Reauthentication interval: 3600 seconds
Supplicant timeout: 30 seconds
Server timeout: 30 seconds
Maximum EAPOL requests: 2
Guest VLAN member: not configured
Number of connected supplicants: 1
  Supplicant: anon@example.net, 52:54:00:BD:8C:E8
    Radius supplicant: user01@example.net
    Operational state: Authenticated
    Backend Authentication state: Idle
    Authentication method: Radius
    Authenticated VLAN: vlan1099
    Session Reauth interval: 3600 seconds
    Reauthentication due in 3486 seconds
    Session Accounting Interim  Interval: 36000 seconds
    Accounting Update  due in 35886 seconds
    Eapol-Block: Not In Effect
    Domain: Data
```

EAP-TTLS Authentication of a Wireless Client

In this example, we use EAP-TTLS as an 802.1X-based authentication of wireless clients.

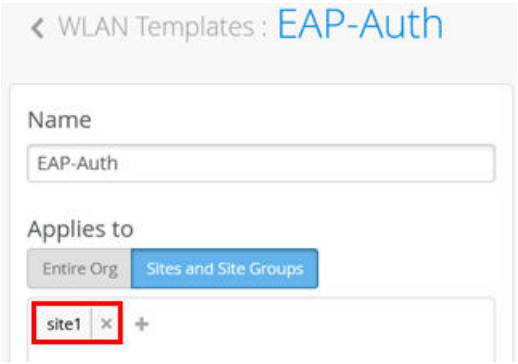
An SSID for wireless can be configured in several ways. In our example, we use a WLAN template by first navigating to **Organization > WLAN Templates**.



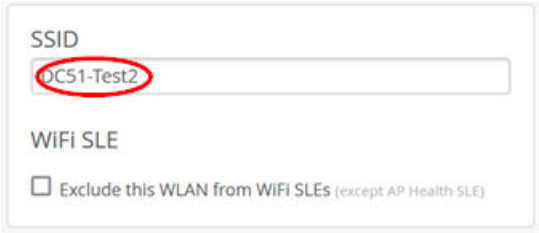
Create a new template with the following settings:

- Name=`EAP-Auth`

- Applies to:
  - Sites and Site Groups=`site1`



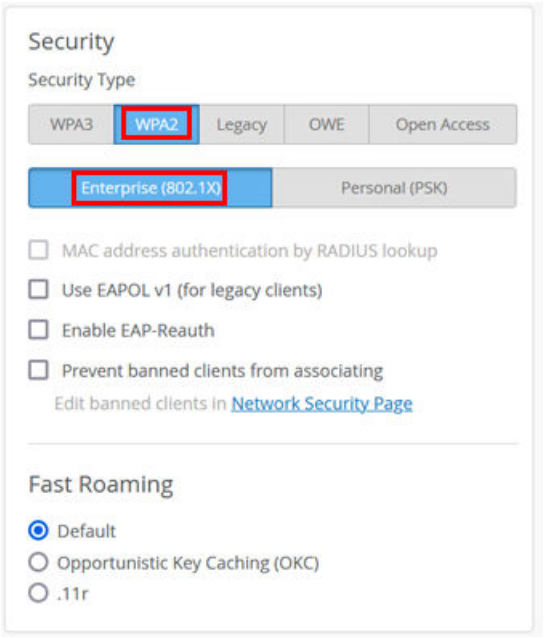
Create an SSID with a name like the figure shown below:



Under **Security,** configure the following settings:

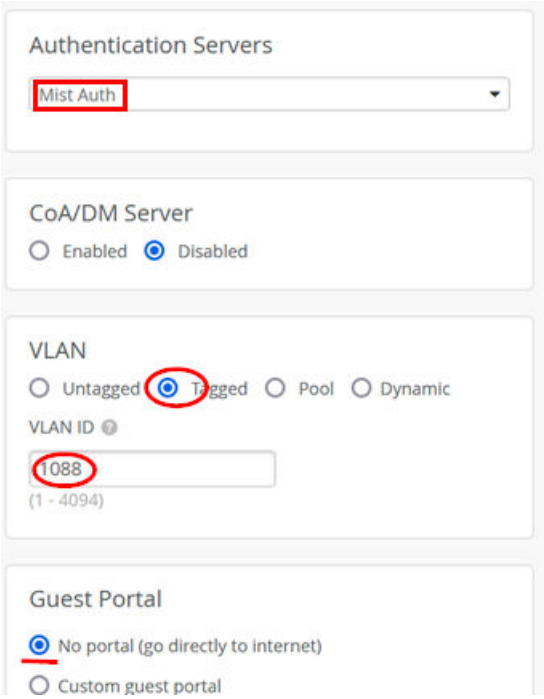
- Security Type=`WPA2` and

```
Enterprise
(802.1X)
```

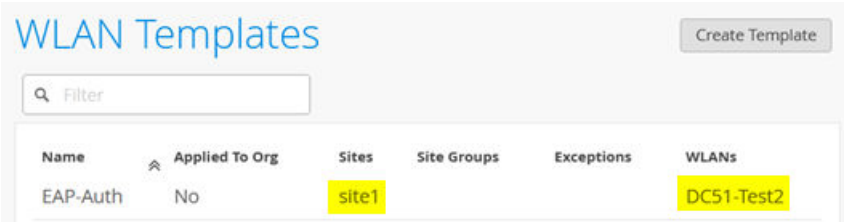


Then, configure the following settings:

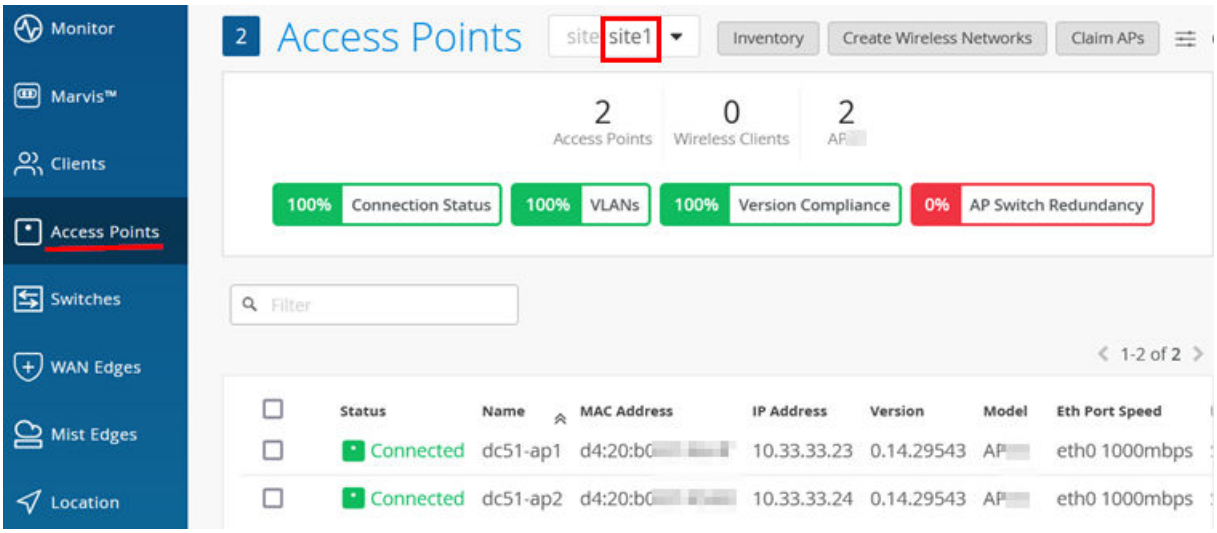
- Authentication Servers=`Mist Auth`
- VLAN=`Tagged`
- VLAN ID=`1088`



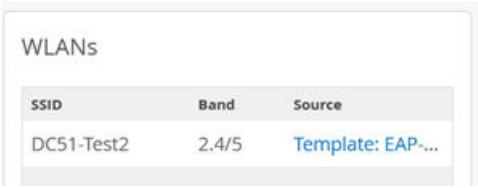
After saving your template, you should see the following configuration:



Next, go **Access Points** and select your site to review the APs. Select one AP.

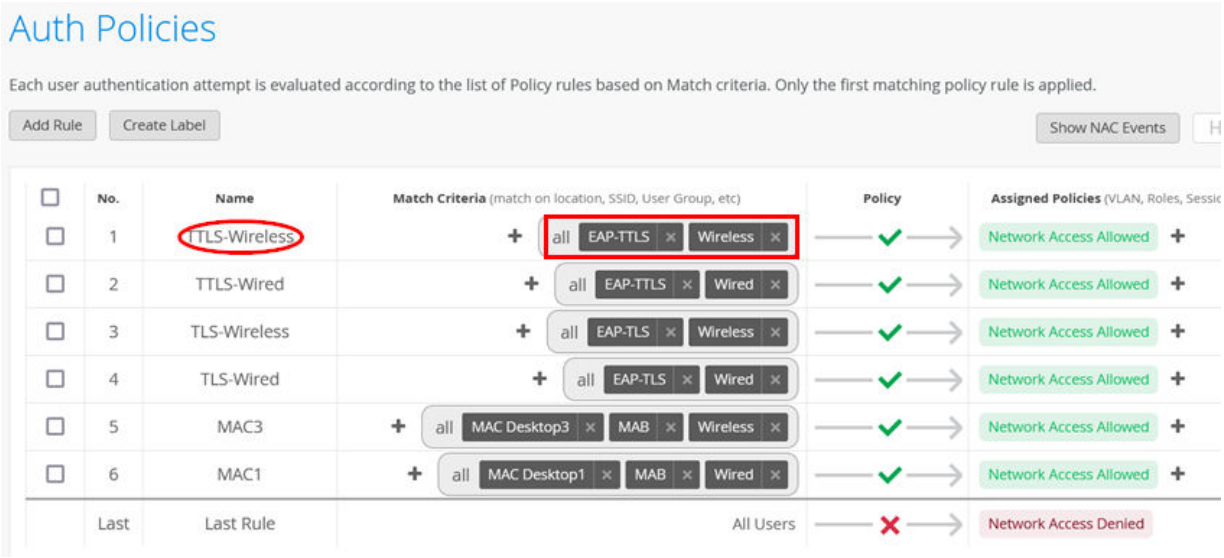


Review the AP configuration applied making sure the SSID from the template appears in the WLANs tab:



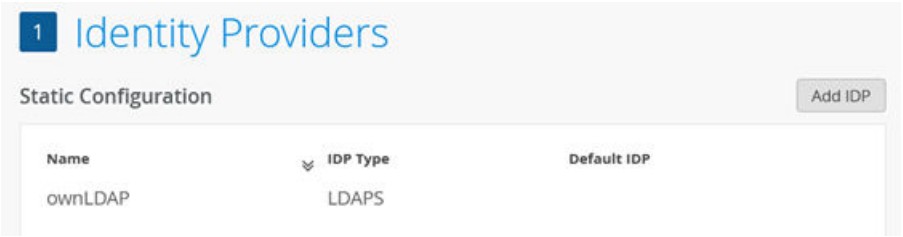
We do not need to specify a label as we only use the authentication type and the port location for identification of the client in this example. So, go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`TTLS-Wireless`
- Match Criteria=`EAP-TTLS` and `Wireless`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



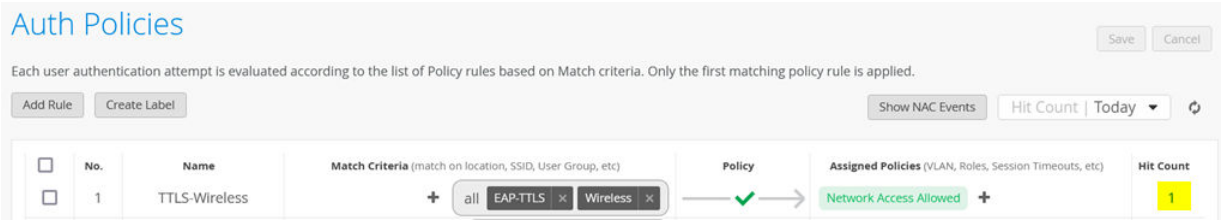
If you have not done it already, you must perform the enterprise PKI integration and let the Juniper Mist authentication cloud learn the root CA and install a TLS server certificate/key for the RADIUS server. Refer to the examples in the section "Juniper Mist Authentication Cloud Certificate Installation" on page 114.

For this test, it is mandatory to have at least one IdP specified since we need to perform a credential check, Hence, the RADIUS server needs to be able to contact a credential database. In our example, we leverage a simplistic LDAP repository. In a production-grade environment, you would probably use Azure or Okta instead. Remember that we have provided examples for those integrations in the section "Configuration Examples of Public Identity Provider Database Integration" on page 154.

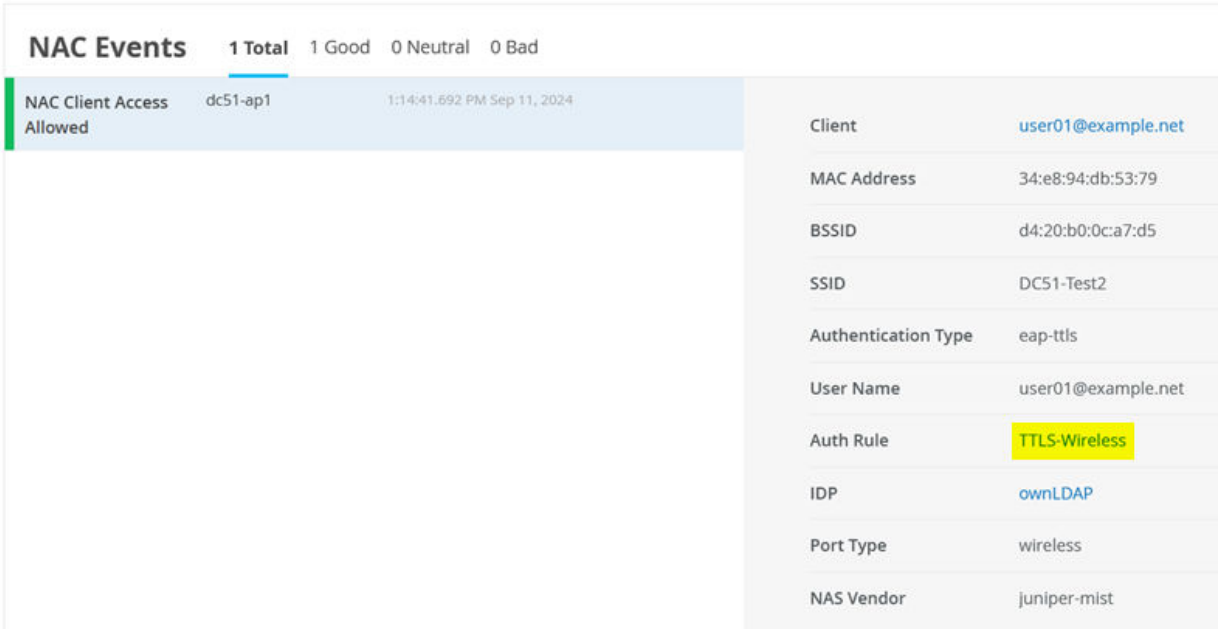


Next, you need to perform EAP-TTLS authentication with a wireless supplicant relevant to your client operating system. We have already shared examples of such configurations for Windows and Linux clients in the section "Configure Client Supplicants with Certificates and Necessary EAP Methods" on page 119.

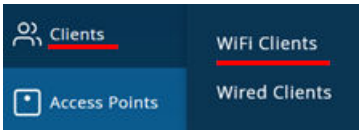
Upon successful completion of the EAP-TTLS authentication, you should see the **Hit Count** increment as in the figure shown below:



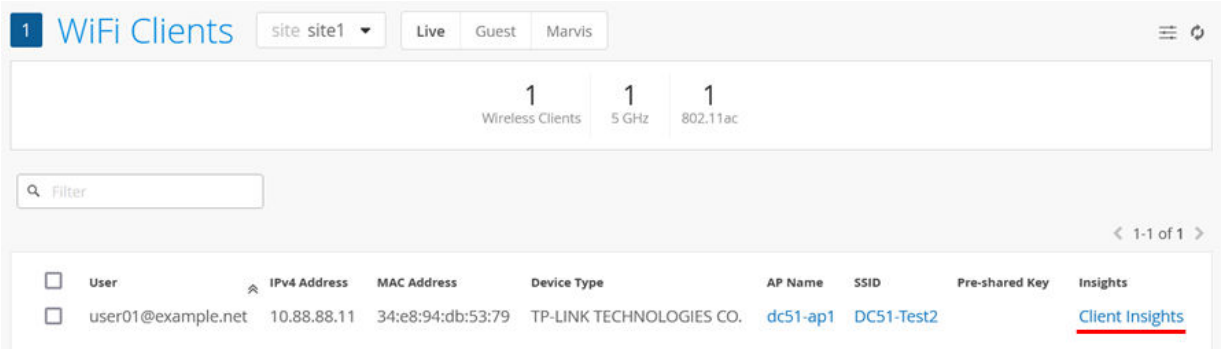
When you click on this link, you will see information about the EAP-TTLS authentications performed.



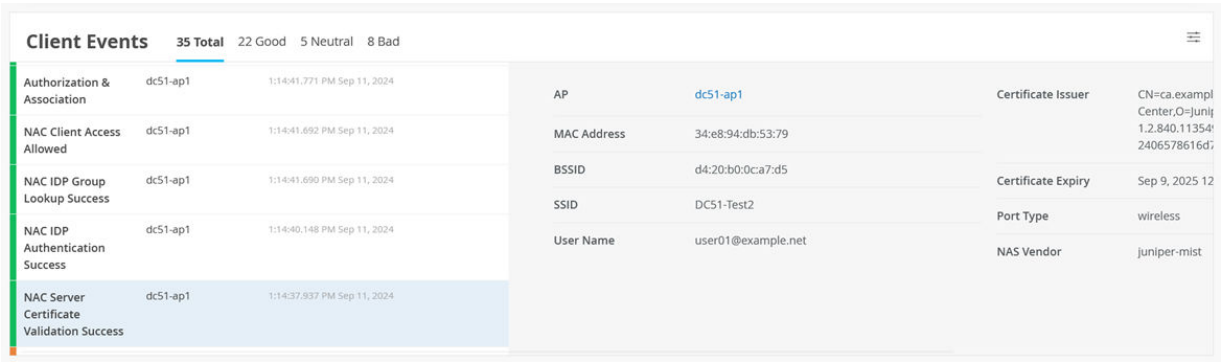
Another way to review information about the wireless client is to go to **Clients > WiFi Clients.**



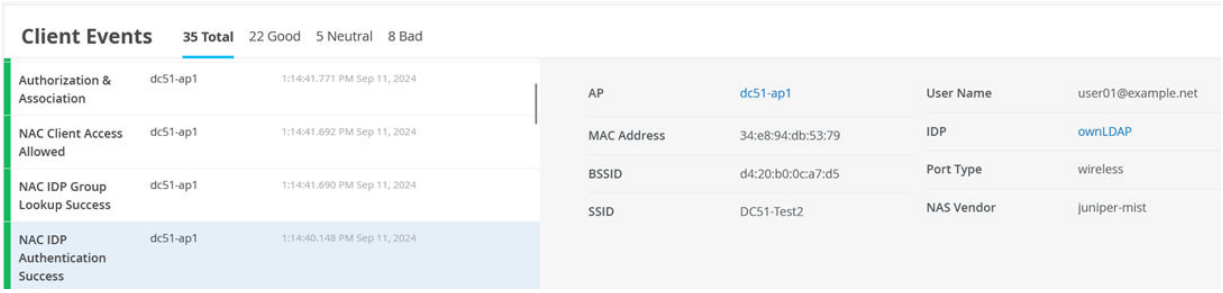
You should see your client listed and where it is attached. Click on **Client Insights**.



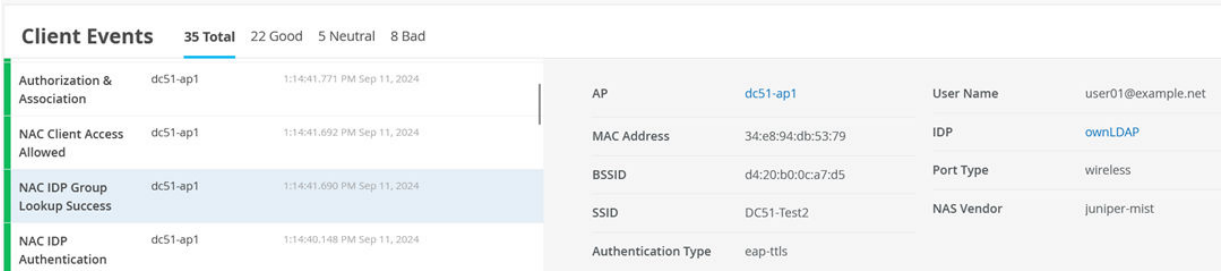
In the **Client Events** section, you will see information about the EAP-TTLS-based authentication process. For instance, the **NAC Server Certificate Validation Success** event.



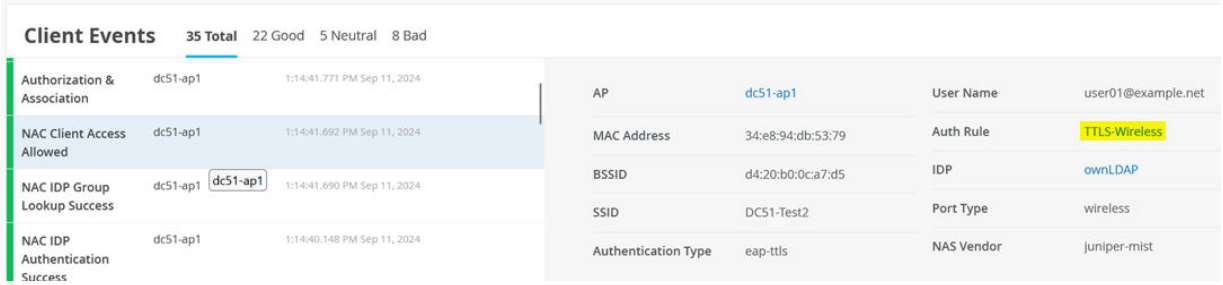
Also see **NAC IDP Authentication Success** events.



And **NAC IDP Group Lookup Success** events.



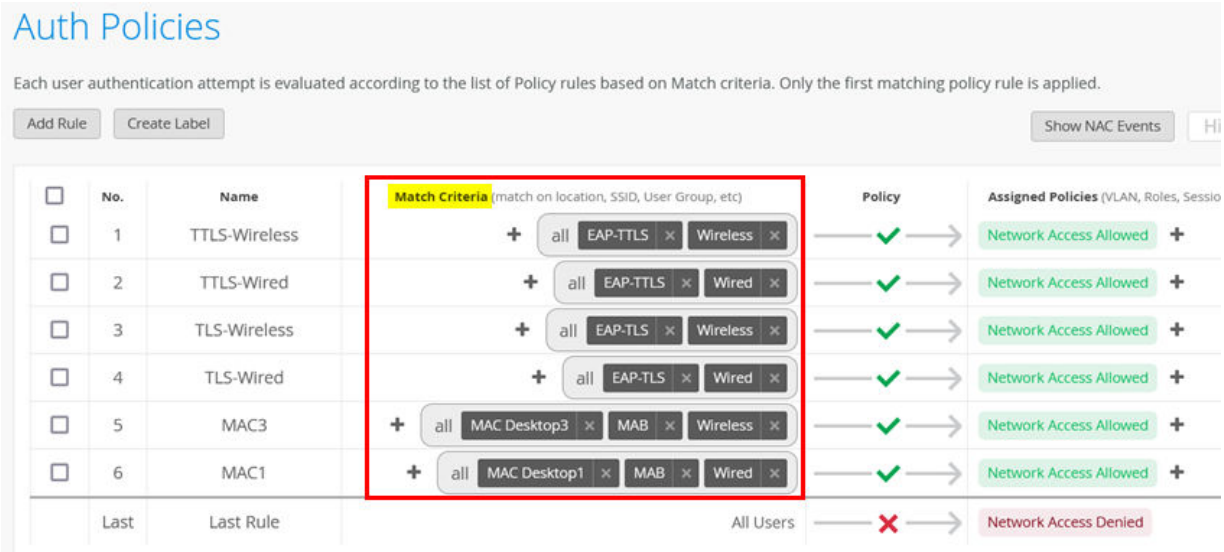
And **NAC Client Access Allowed** events.



Policy Match Criteria Checking

When creating an authentication policy, you must define one or more match criteria for the evaluation of the policy to be performed. For a certain policy, all defined match criteria must be present to match the policy. They are defined as logical AND conditions. During the authentication tests, a list such as that shown in Figure 33 on page 233 was created.

Figure 33: Authentication Testcase Policies



There are, however, more match criteria available than shown in the figure above. The complete list of all available match criteria is:

- Authentication type:
  - MAB (MAC address-based authentication)
  - EAP-TLS
  - EAP-TTLS
  - TEAP
  - PSK (for wireless only)
  - Admin Auth
- Port type (over which the client authentication is performed):
  - Wired
  - Wireless

- RADIUS attribute-based (you can also use an auth label definition for these):
  - Check the vendor list for RADIUS AVPs
- Sites and site groups (location)
- Auth label-based (provides the most flexibility):
  - Certificate attribute—Checks the supplicant attributes. Note: EAP-TTLS does not support this as it does not use client certificates.
  - Client list—MAC addresses or OUI definitions when doing MAB.
  - Directory attribute—Requires integration with an IdP database.
  - SSID—Cannot be used for wired clients.
  - MDM Compliance—This requires integration with an IdP database and an MDM.
  - Client Label—These labels can be assigned when a wireless client uses a Juniper AP.

In the authentication test cases, we used the following match criteria with reference to Figure 33 on page 233:

- Authentication Type=`MAB` and `EAP-TLS` and `EAP-TTLS`
- Port Type=`Wired` and `Wireless`
- Auth Label=`Client List` with two different MAC addresses

We added an example of match criteria in this section and will demonstrate how to properly implement certificate attribute checking for EAP-TLS.

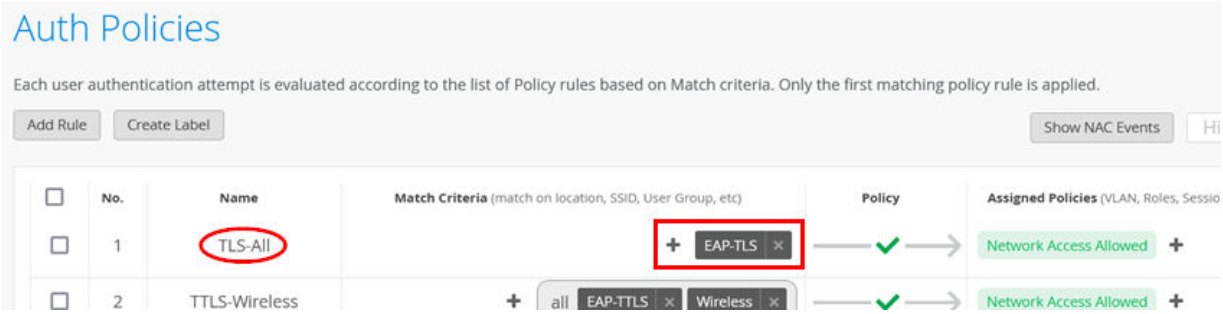
First, we need to determine which certificate attributes are used by the supplicant when it authenticates with EAP-TLS. Two possible methods to determine this information are as follows:

- Obtain the information when reviewing the certificate after it's installed on the supplicant.
- The recommended method is:
  - Create a generic EAP-TLS authentication policy.
  - Perform a successful EAP-TLS authentication with your client.
  - Review the **Client Events** and check the certificate attributes that are logged.
  - Create an auth label based on these certificate attributes.
  - Create a new authentication policy (with a priority over the generic policy) using the new auth label.
  - Perform the EAP-TLS authentication with your client again.

- Confirm that the more specific authentication policy is being matched instead.

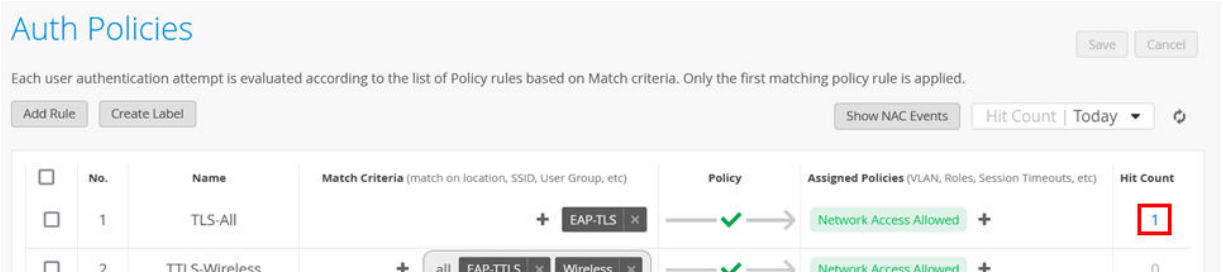
We begin with the generic rule. Go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`TLS-All`
- Match Criteria=`EAP-TLS`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



Now, perform any EAP-TLS client authentication using either wired or wireless.

After the authentication has been performed, you should notice that the policy **Hit Count** has increased.

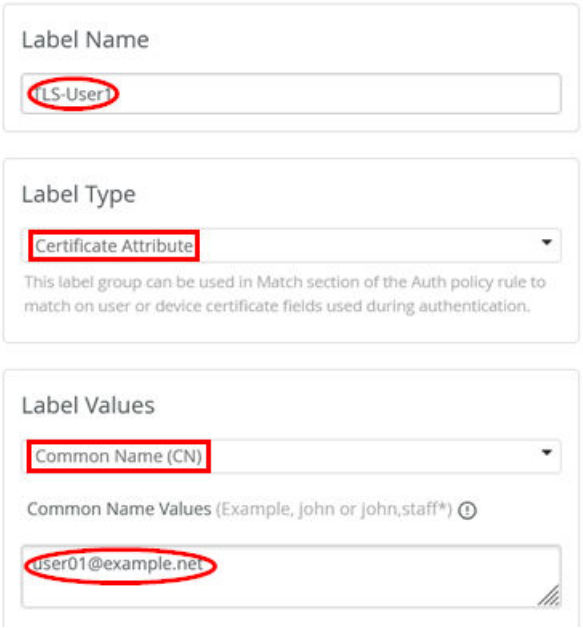


Among other logged information, you will now see various information about the client certificate used on the supplicant.

Authentication Type	eap-tls
User Name	user01@example.net
Certificate CN	user01@example.net
Certificate SAN (Email)	user01@example.net
Certificate Issuer	/C=NL/ST=Netherlands/L=Amsterdam/O=Juniper/OU=CA-Center/ CN=ca.example.net/emailAddress=trustcenter@example.net
Certificate Expiry	Sep 9, 2025 12:07:12 PM
Certificate Subject	/C=US/ST=California/O=Example TEST-Corp./OU=Human Resources Dept./ CN=user01@example.net
Auth Rule	TLS-All

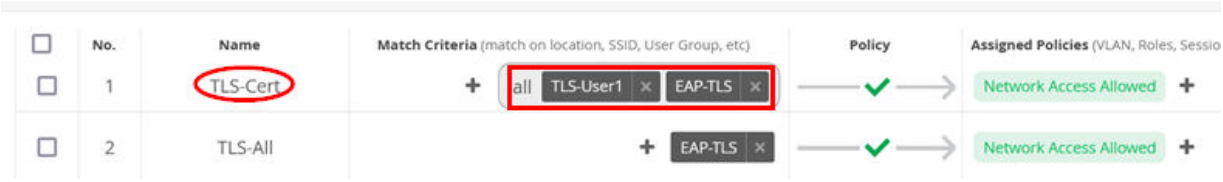
This information allows you to create an auth label specific to the user. In the example, we copy and paste the reported common name (CN) certificate attribute to a new label by navigating to **Organization > Auth Policy Labels**:

- Label Name=`TLS-User1`
- Label Type=`Certificate Attribute`
- Label Values=`Common Name (CN)`
- Common Name Values=`user01@example.net`



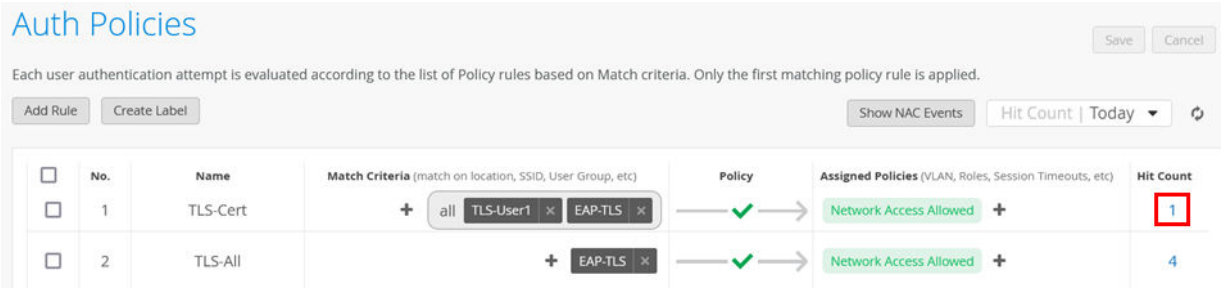
Then, create a more specific rule using the new label and position it above the previous rule. Go to **Organization > Auth Policies** and create the following rule:

- Position=`1`
- Name=`TLS-Cert`
- Match Criteria=`EAP-TLS` and `TLS-User1`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed`



Next, perform a new EAP-TLS client authentication with the client again.

With the right certificate attribute in place, this policy should now get a hit, and the other more generic rule will be used for any other clients not having the expected certificate attribute.



Authorization and Assigned Policies

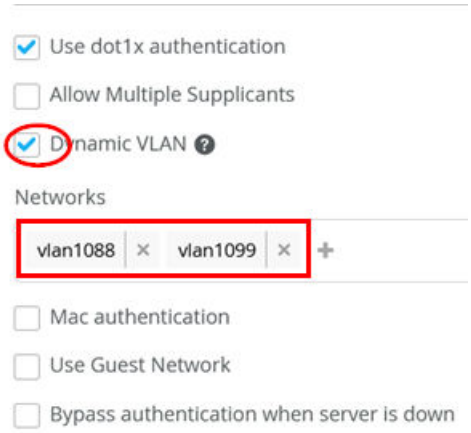
So far, we have not used any assigned policies. However, they are critical for enabling the RADIUS server to not just approve authentication, but in its response, to optionally instruct the network access device (the switch or AP) to enforce any limitations on the network access provided to the attached client. Every client may not have the same network access rights after being authenticated. As a result, we need to be able to enforce different levels of access rights where the client ingresses the network. We achieve this by adding rules when we allow network access to a certain authentication policy.

When defining any authorization parameters, we must do this by creating an auth label:

- Label Type=AAA Attribute
- Label values to choose from:
  - VLAN—Allows you to specify a single VLAN for this client to access. This is a very commonly used option. It usually leverages the standard RADIUS AVP=`Tunnel-Private-Group-ID`
  - GBP Tag—Used in IP Clos fabrics, this allows you to specify the number of a group-based policy (GBP) tag to be dynamically assigned. It leverages the Juniper RADIUS AVP=`Juniper-Switching-Filter` with a string containing the GBP-Tag configuration.
  - Session Timeout—Allows you to shorten or lengthen the time a client is allowed access to the network before a new authentication must be performed.
  - Custom Vendor Specific Attribute—Allows you to use non-standard RADIUS AVPs. You need to know that the vendor of the network access device supports a custom attribute.
  - Custom Standard RADIUS Attribute—Allows you to use standard RADIUS AVPs.
  - Dynamic Wired Port Configuration—Allows you to configure trunk and native VLANs as a list of VLANs to be configured. It usually leverages the standard RADIUS AVP=`Egress-VLAN-Name` multiple times.

NOTE: You may see other RADIUS attributes to select, but they only make sense when the auth label is used for a match criteria. So, ignore “Role”, “Realm”, and “User Name”, for example.

Care must be taken when using RADIUS attributes to configure one or more VLANs that are not configured on the port beforehand. By default, the system does not configure a VLAN at the time it’s needed using a dynamic configuration option. Especially when using campus fabric, there may be a situation where the ports you have defined for an access switch or Virtual Chassis do not use a certain VLAN. Hence, the fabric renderer will not provide a configuration for this VLAN locally to the access switch or Virtual Chassis and when you configure a VLAN as an authorization parameter, it’s not actually configured on the port. In this case, you must use the `Dynamic VLAN` option on the port profile and add all potential VLANs you might use for future dynamic assignments, as described in the example below. This will instruct the fabric renderer to configure those VLANs ahead of time before you can use them dynamically via RADIUS.



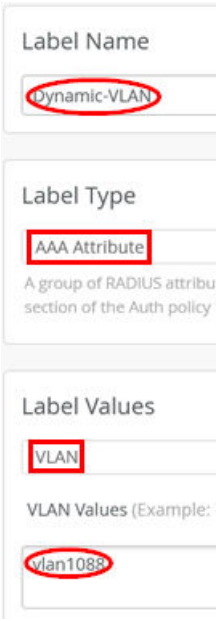
Assigned Policy of a Single Dynamic VLAN

In this example, upon authentication, we dynamically change the local VLAN a client gets assigned to. Review the test case for "EAP-TLS Authentication of a Wired Client" on page 204. You will find that at the end of that process, the client was assigned to `vlan1099` because this was the default VLAN in that port profile.

```
root@switch1> show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
  Supplicant timeout: 30 seconds
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 1
    Supplicant: user01@example.net, 52:54:00:BD:8C:E8
      Operational state: Authenticated
      Backend Authentication state: Idle
      Authentication method: Radius
      Authenticated VLAN: vlan1099
      Session Reauth interval: 3600 seconds
      Reauthentication due in 3287 seconds
      Session Accounting Interim  Interval: 36000 seconds
      Accounting Update  due in 35687 seconds
      Eapol-Block: Not In Effect
      Domain: Data
```

Now let's assume that we want to have vlan1088 assigned to the client instead. In this case, define an auth label for this by first navigating to **Organization > Auth Policy Labels** and configuring the following settings:

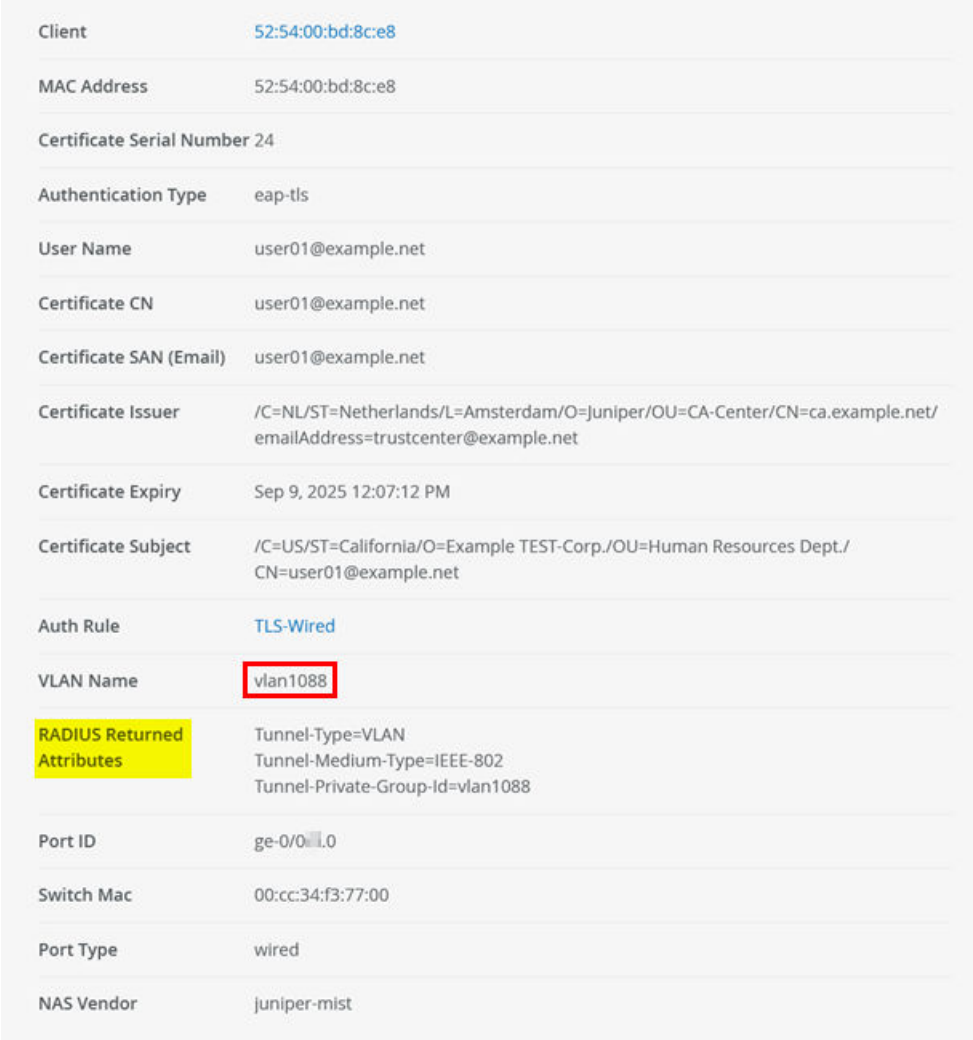
- Label Name=`Dynamic-VLAN`
- Label Type=`AAA Attribute`
- Label Values=`VLAN`
- `VLAN Values=vlan1088`



Then, change the existing auth policy rule to add our label to the existing `TLS-Wired` auth policy like in the figure shown below:

Next, perform EAP-TLS client authentication for your wired client.

After the new authentication is successful, check the policy hit and confirm that the dynamic VLAN is used instead as indicated in the figure below:



Client	52:54:00:bd:8c:e8
MAC Address	52:54:00:bd:8c:e8
Certificate Serial Number	24
Authentication Type	eap-tls
User Name	user01@example.net
Certificate CN	user01@example.net
Certificate SAN (Email)	user01@example.net
Certificate Issuer	/C=NL/ST=Netherlands/L=Amsterdam/O=Juniper/OU=CA-Center/CN=ca.example.net/ emailAddress=trustcenter@example.net
Certificate Expiry	Sep 9, 2025 12:07:12 PM
Certificate Subject	/C=US/ST=California/O=Example TEST-Corp./OU=Human Resources Dept./ CN=user01@example.net
Auth Rule	TLS-Wired
VLAN Name	vlan1088
RADIUS Returned Attributes	Tunnel-Type=VLAN Tunnel-Medium-Type=IEEE-802 Tunnel-Private-Group-Id=vlan1088
Port ID	ge-0/0[redacted].0
Switch Mac	00:cc:34:f3:77:00
Port Type	wired
NAS Vendor	juniper-mist

(Optional) You can also see the effects when opening a Remote Shell on the switch:

```
              show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
```

```
  Supplicant timeout: 30 seconds
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 1
    Supplicant: user01@example.net, 52:54:00:BD:8C:E8
      Operational state: Authenticated
      Backend Authentication state: Idle
      Authentication method: Radius
      Authenticated VLAN: vlan1088
      Session Reauth interval: 3600 seconds
      Reauthentication due in 3592 seconds
      Session Accounting Interim  Interval: 36000 seconds
      Accounting Update  due in 35992 seconds
      Eapol-Block: Not In Effect
      Domain: Data
show vlans vlan1088 detail
Routing instance: default-switch
VLAN Name: vlan1088                      State: Active
Tag: 1088
Internal index: 3, Generation Index: 3, Origin: Static
MAC aging time: 300 seconds
VXLAN Enabled : No
Interfaces:
    ge-0/0/3.0*,tagged,trunk
    ge-0/0/0.0*,untagged,access
Number of interfaces: Tagged 1    , Untagged 1
Total MAC count: 1
```

Assigned Policy for Multiple VLANs on a Trunk Port and AP as Supplicant

In this example, we demonstrate a few features at the same time:

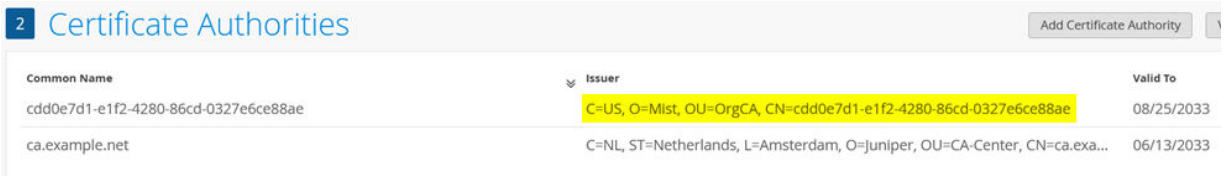
- The capability to reuse the automatically generated and deployed certificate, which the AP uses to authenticate the RadSec tunnel connecting to the Juniper Mist cloud, as a certificate for itself as a supplicant.
- The capability of the Juniper AP to function as an 802.1X supplicant by utilizing a device certificate for EAP-TLS authentication with a switch.

- The ability to dynamically assign all required trunk VLANs for connected clients, as well as the AP's native management VLAN, upon successful authentication of the AP.

The benefit of this feature combination allows us to not only strongly authenticate the clients on the network but also the attached infrastructure, including APs. Without such protection of switch ports, an attacker could disable an AP, attach its ethernet cable to their own laptop or AP and access any VLAN configured on the port. An additional advantage of this feature combination is the increased flexibility in using switch ports for various devices and clients. By enabling EAP 802.1X authentication on all switch ports, the appropriate VLANs are automatically assigned to a port when a recognized client or AP is connected. This avoids static configuration and the reservation of precious infrastructure switch ports.

NOTE: The supplicant feature on Juniper APs is not supported by the models AP21, AP41 and AP61 which have been announced for EOL. All other models need firmware 0.14.x or higher.

The first step of this configuration is to remember that as part of the configuration in section "Juniper Mist Authentication Cloud Certificate Installation" on page 114, we have configured the Mist CA as an additional root CA. We need to extract the issuer to be able to identify the certificates the AP will use for authentication.

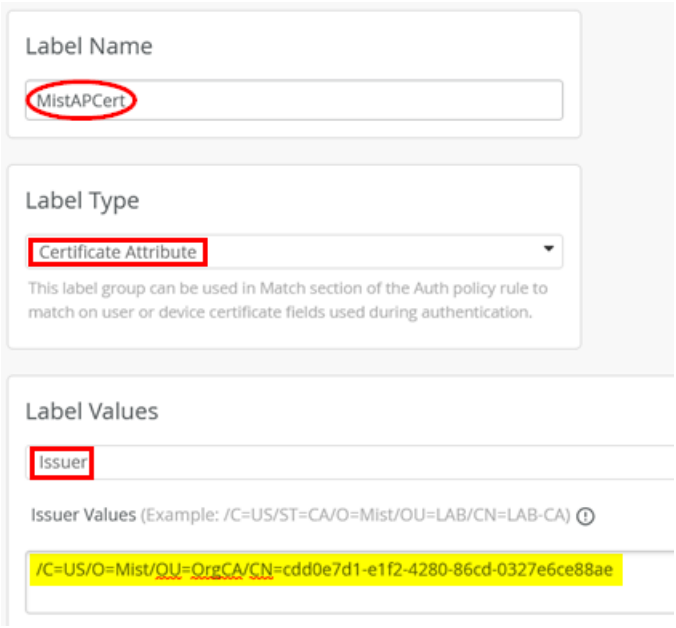


From the extracted issuer, you need to delete all of the space characters and commas, then use / as a new delimiter. In our example, the result looks like:

```
/C=US/O=Mist/OU=OrgCA/CN=cdd0e7d1-e1f2-4280-86cd-0327e6ce88ae
```

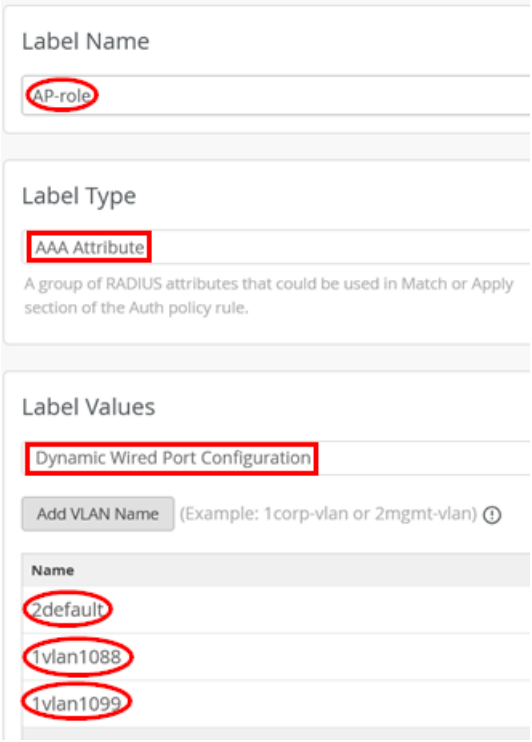
With that information, define a new label with the following settings by navigating to **Organization > Auth Policy Labels**:

- Label Name=`MistAPCert`
- Label Type=`Certificate Attribute`
- Label Values=`Issuer`
- Issuer Values=`<insert your own issuer>`



Continue with the next label that defines which client VLANs this AP will use:

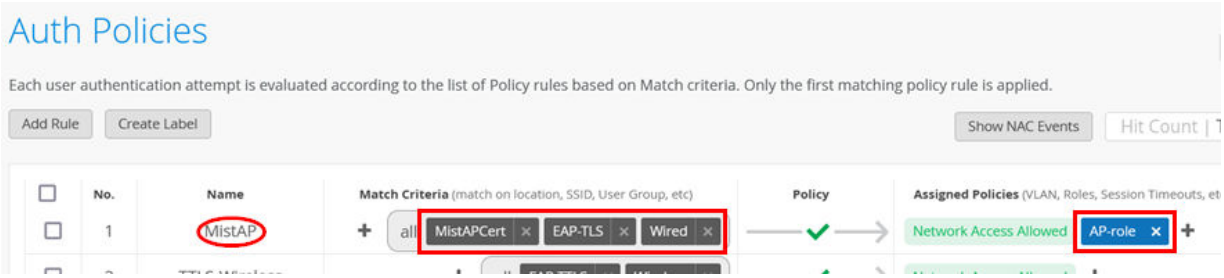
- Label Name=`AP-role`
- Label Type=`AAA Attribute`
- Label Values=`Dynamic Wired Port Configuration`
  - Name1=`2default`
  - Name2=`1vlan1088`
  - Name3=`1vlan1099`



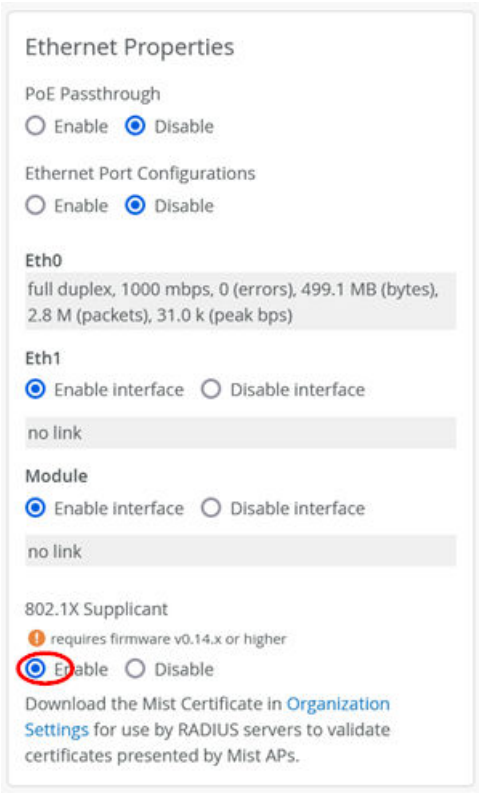
NOTE: When entering the VLAN names, it is critical to know that the first character defines whether the VLAN needs to be tagged or not. Use “2” for the native, untagged VLAN that is usually used to manage the AP itself and use “1” for tagged VLANs.

Next, create the necessary auth policy. Go to **Organization -> Auth Policies** and create the following rule:

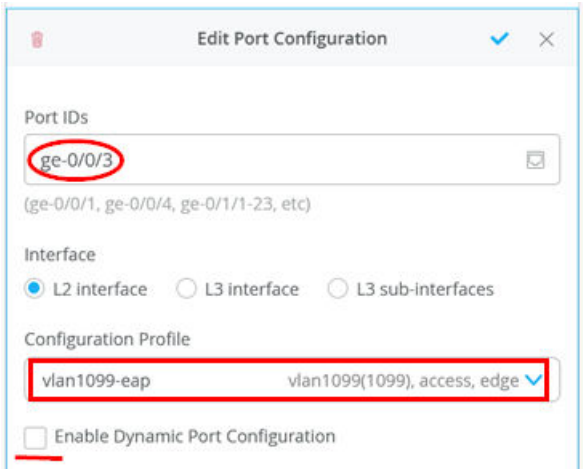
- Position=`1`
- Name=`MistAP`
- Match Criteria=`MistAPCert` and `EAP-TLS` and `Wired`
- Policy Pass=`Pass`
- Assigned Policies=`Network Access Allowed` and `AP-role`



Next, we must enable the AP as a supplicant using the same certificate it uses for its RadSec tunnel towards the Juniper Mist cloud. To do so, go to **Access Points > <your AP>** and enable the 802.1X supplicant under **Ethernet Properties** as shown in the figure below:



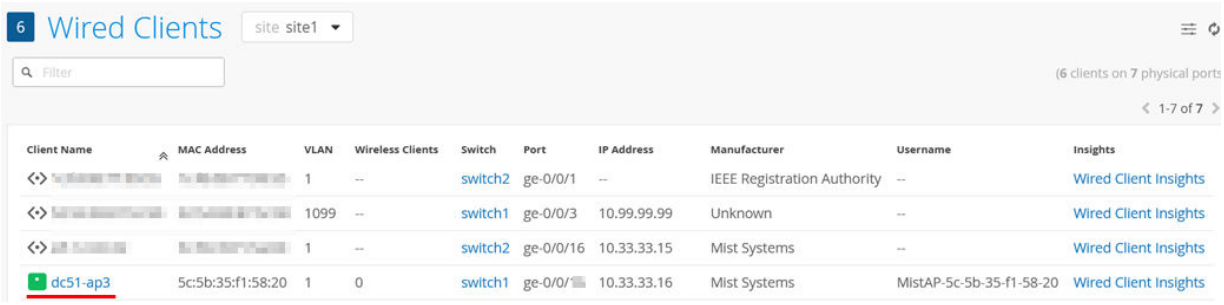
Next, change the AP-attached port on the access switch to use the configuration profile defined for 802.1X in Figure 26 on page 84 in the switch template. Change the configuration profile to `vlan1099-eap`.



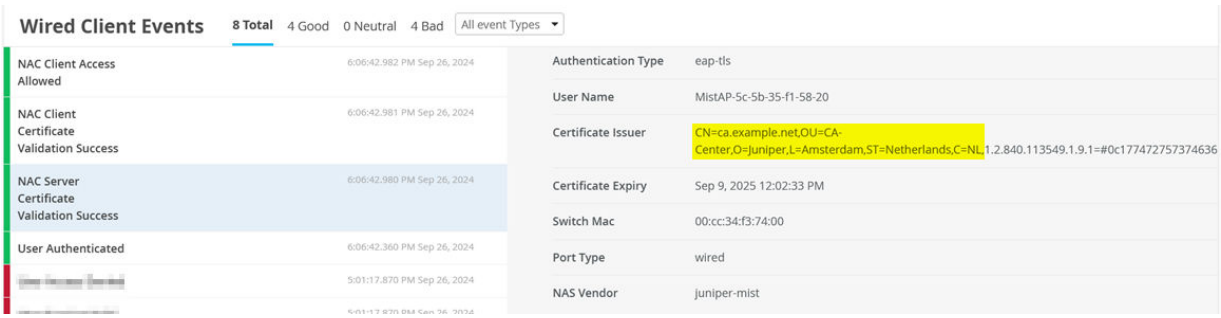
NOTE: The RADIUS-based dynamic VLAN configuration does not require enabling the checkbox you see for **Dynamic Port Configuration** in the figure above. Leave this unchecked since it's meant for a different kind of dynamic port configuration the Juniper switch supports.

Next, the AP should authenticate through the switch using EAP-TLS and receive a dynamically assigned VLAN through RADIUS.

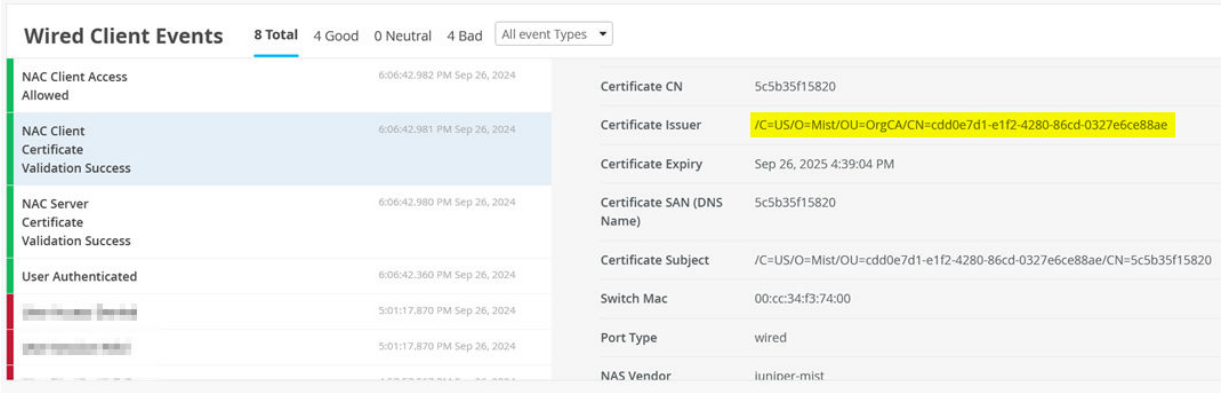
You should see the AP as a wired client:



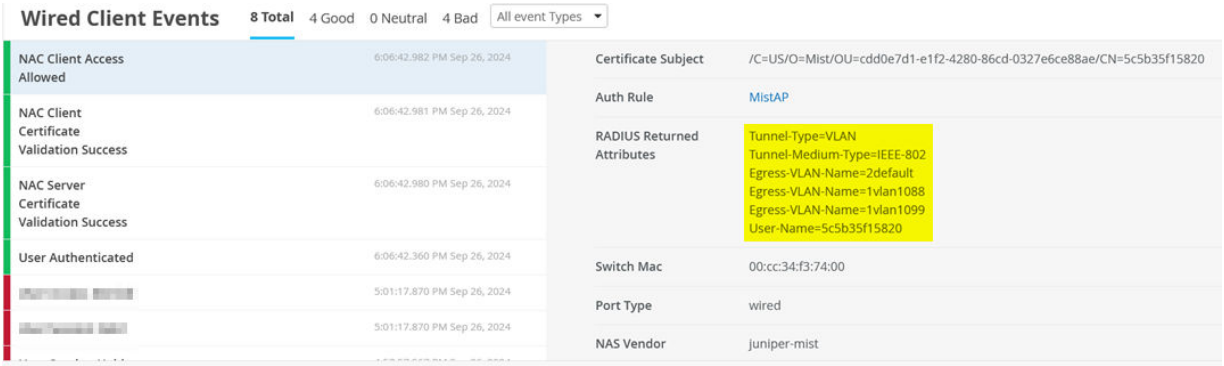
You can see that the issuer of the RADIUS certificate is your own organization's PKI:



The AP client certificate should get validated:



Finally, the AP is allowed to access the network and the other VLANs are assigned to this port through the auth rule and dynamic VLAN assignments:



(Optional) Using Remote Shell, you should see the following information on the switch:

```
root@access1> show dot1x interface ge-0/0/3
802.1X Information:
Interface     Role           State           MAC address          User
ge-0/0/3.0    Authenticator  Authenticated   5C:5B:35:F1:58:20    5c5b35f15820
root@access1> show dot1x interface ge-0/0/3 detail
ge-0/0/3.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
```

```
  Supplicant timeout: 30 seconds
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 1
    Supplicant: MistAP-5c-5b-35-f1-58-20, 5C:5B:35:F1:58:20
      Radius supplicant: 5c5b35f15820
      Operational state: Authenticated
      Backend Authentication state: Idle
      Authentication method: Radius
      Authenticated VLAN: default
      Session Reauth interval: 3600 seconds
      Reauthentication due in 3530 seconds
      Egress Vlan: 1, 1088, 1099
      Session Accounting Interim  Interval: 36000 seconds
      Accounting Update  due in 35930 seconds
      Eapol-Block: Not In Effect
      Domain: Data
root@access1> show vlans
Routing instance        VLAN name             Tag          Interfaces
default-switch          default               1
                                                           ae0.0*
.
                                                           ge-0/0/3.0*
.
default-switch          vlan1088              1088
                                                           ae0.0*
                                                           ge-0/0/3.0*
default-switch          vlan1099              1099
                                                           ae0.0*
                                                           ge-0/0/0.0*
                                                           ge-0/0/3.0*
```

Assigned Policy by Referencing a Filter-ID

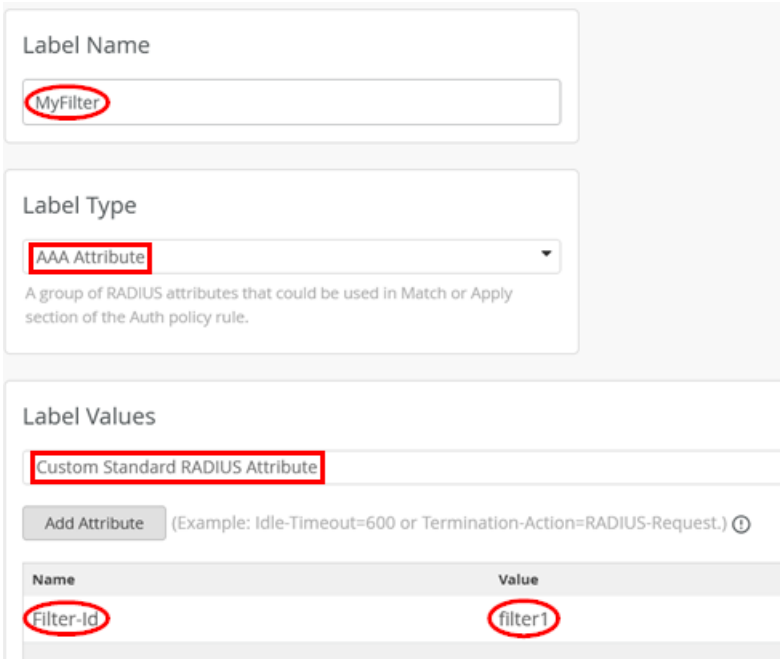
When using the standard RADIUS AVP `Filter-Id`, you can dynamically assign EX Switch firewall filters to wired clients upon their authentication. This capability and the configuration options of EX Switch firewall filters are explained here.

For this test case, first apply a minimalistic firewall filter that enables policing and counting packets for a dynamic wired client. We recommend applying this configuration to all switches at a site using a switch template. Alternatively, you can apply the below ruleset locally using additional Junos OS CLI commands to the switch.

```
delete firewall policer p1
set firewall policer p1 if-exceeding bandwidth-limit 1m
set firewall policer p1 if-exceeding burst-size-limit 2k
set firewall policer p1 then discard
delete firewall family ethernet-switching filter filter1
set firewall family ethernet-switching filter filter1 term t1 from source-address 10.99.99.0/24
set firewall family ethernet-switching filter filter1 term t1 then count counter1
set firewall family ethernet-switching filter filter1 term t1 then policer p1
set firewall family ethernet-switching filter filter1 term t1 then log
```

The next step is to create a new label by going to **Organization > Auth Policy Labels** and configuring the following settings:

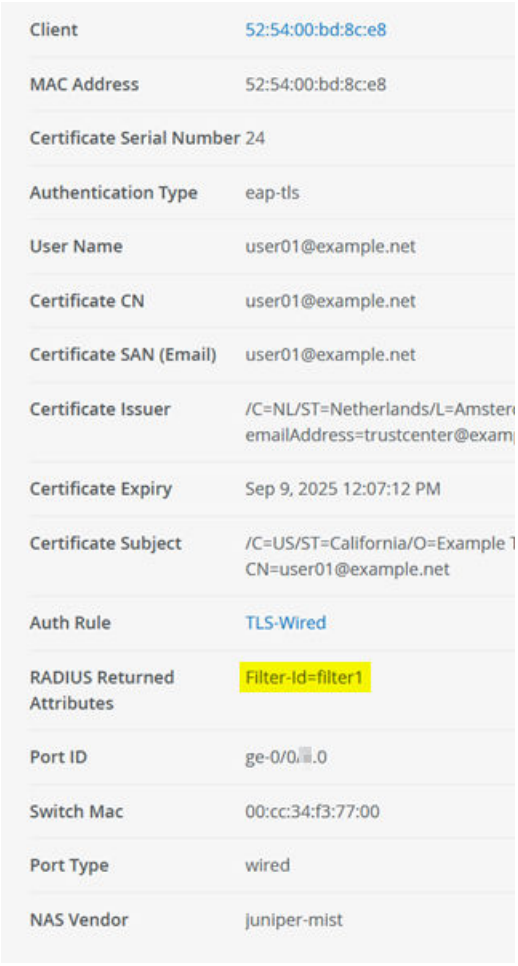
- Label Name=`MyFilter`
- Label Type=`AAA Attribute`
- Label Values=`Custom Standard RADIUS Attribute`
- Name1=`Filter-Id`
- Value1=`filter1`



Then, change the existing auth policy rule to add the label to the existing `TLS-Wired` auth policy like in the figure shown below:

Next, perform EAP-TLS client authentication for the wired client.

After the client authenticates, check the policy hit and confirm that the filter-ID attribute was used as indicated in the figure below:



(Optional) Remote Shell to the switch and execute the two commands shown in the figure below to confirm that the filter was applied correctly and works as expected:

```
              show dot1x interface ge-0/0/0 detail
ge-0/0/0.0
  Role: Authenticator
  Administrative state: Auto
  Supplicant mode: Single
  Number of retries: 3
  Quiet period: 60 seconds
  Transmit period: 30 seconds
  Mac Radius: Disabled
  Mac Radius Restrict: Disabled
  Reauthentication: Enabled
  Reauthentication interval: 3600 seconds
  Supplicant timeout: 30 seconds
```

```
  Server timeout: 30 seconds
  Maximum EAPOL requests: 2
  Guest VLAN member: not configured
  Number of connected supplicants: 1
    Supplicant: user01@example.net, 52:54:00:BD:8C:E8
      Operational state: Authenticated
      Backend Authentication state: Idle
      Authentication method: Radius
      Authenticated VLAN: vlan1099
      Dynamic Filter: filter1
      Session Reauth interval: 3600 seconds
      Reauthentication due in 3423 seconds
      Session Accounting Interim  Interval: 36000 seconds
      Accounting Update  due in 35823 seconds
      Eapol-Block: Not In Effect
      Domain: Data
show dot1x firewall
Filter name: dot1x_ge-0/0/0
Counters:
Name                                                Bytes              Packets
counter1__dot1x_ge-0/0/0-filter1-t1                  3430                   34
Filter name: dot1x_ge-0/0/0
Policer:
Name                                                                 Packets
p1-t1                                                                     33
```

Revision History

Table 4: Revision History

Date	Version	Description
February 2025	JVD-ENTWIRED-MAA-01-01	Initial publish