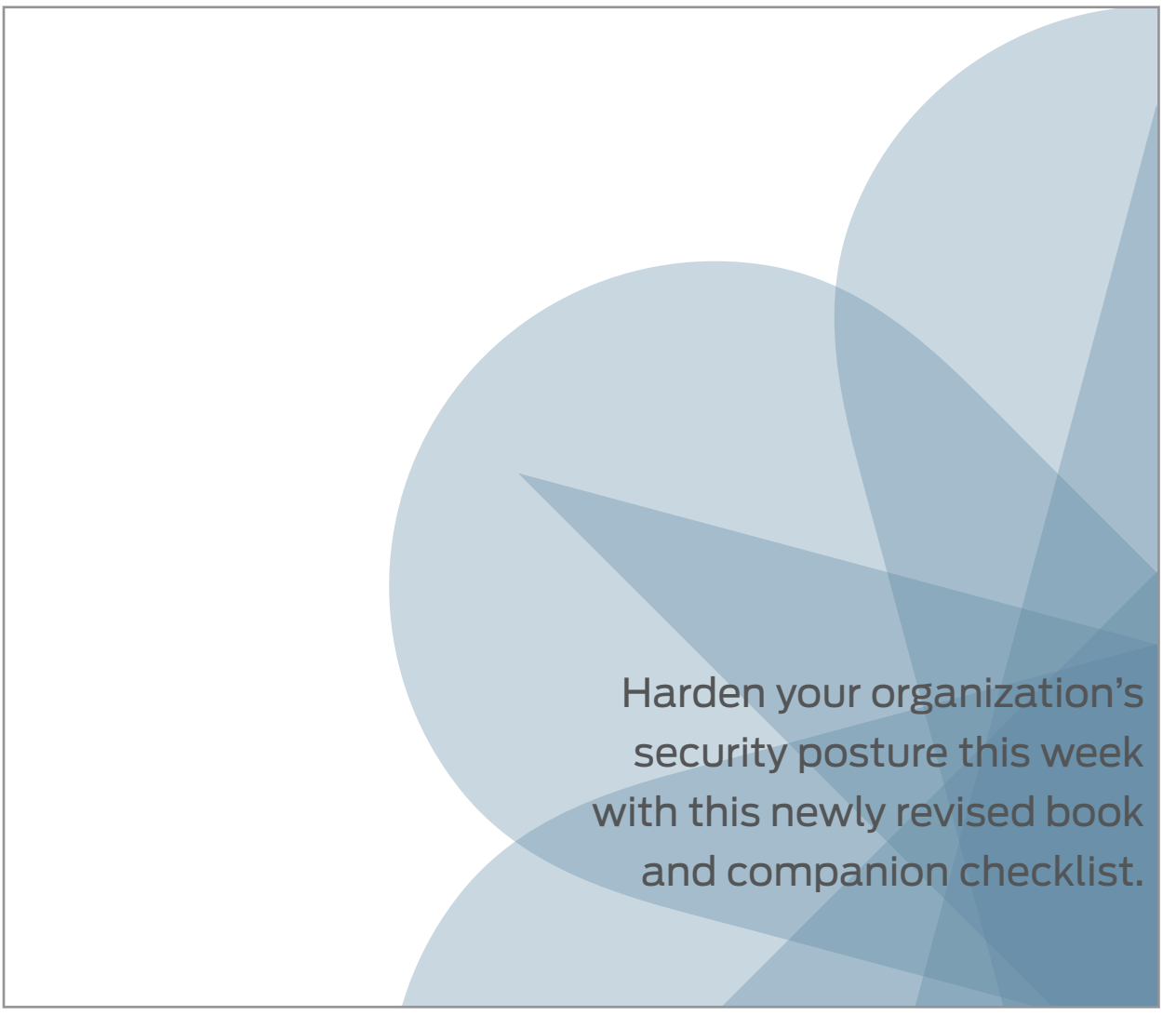
JUNIPER NETWORKS

THIS WEEK: HARDENING JUNOS DEVICES, 2ND EDITION

The best-selling book now updated and revised!



By John Weidley

THIS WEEK: HARDENING JUNOS DEVICES, 2ND EDITION

Juniper Networks takes the security of its products very seriously and has created proven processes and procedures following industry best practices. *This Week: Hardening Junos Devices, 2nd Edition* divides Juniper's hardening procedures into four topic areas – Non-Technical, Physical Security, Operating System Security, and Configuration Hardening – and delves into sample strategies, example configurations, and dozens of suggestions and useful tips for implementing each hardening process. All features discussed in this book are available and tested in **Junos 12.3** (current recommended code) and for some features the book discusses options available in later code releases.

Encyclopedic in its coverage, *This Week: Hardening Junos Devices, 2nd Edition* is a book you cannot afford not to read. The author's 15 years of experience supporting U.S. Government agencies makes it applicable to high security environments such as service providers, financial institutions, government, and enterprise networks. But it's also pertinent to the devices in your wiring closet and branch office. Once you take care of the physical security, you can harden your Junos device to resist attacks and diversions, as well as the careless mishaps that haunt even the most experienced network engineer. This book also includes a handy checklist you can print or copy for each device you control.

> "The best network design will not help you if you forget to thoroughly secure and harden your network devices. This book is particularly welcomed by those taking their first steps into the Junos world - it helps map concepts from Cisco IOS into various Junos dialects as well as covering all the bits and pieces you might never even consider, like securing the LCD menu."
>
> *Ivan Pepelnjak, Network architect, ipSpace.net AG, www.ipSpace.net*

LEARN HOW TO HARDEN YOUR SECURITY POSTURE THIS WEEK:

- Review the non-technical aspects of device management that are so critical to the overall security posture of your organization.
- Understand how physical security is an important aspect of device deployment.
- Understand and deploy the Junos operating system's inherent security features.
- Identify important management, access services, and user account restrictions to provide least privileged access.
- Configure route authentication for popular routing and signaling protocols.
- Create and apply a firewall filter to protect the routing engine.

ISBN 978-1941441190

Published by Juniper Networks Books
www.juniper.net/books

JUNIPER NETWORKS

This Week: Hardening Junos Devices, 2nd Edition

By John Weidley

JUNIPER NETWORKS

Published by Juniper Networks Books
Author: John Weidley
Technical Reviewers: Tim Brown and Richard Woodman
Editor in Chief: Patrick Ames
Copyeditor and Proofer: Nancy Koerbel
Illustrator: Karen Joice
J-Net Community Manager: Julie Wider

About the Author
John Weidley is a Resident Engineer with Juniper Networks. He has been certified in Juniper Networks as JNCIS-SEC, JNCIS-SSL, JNCIA-FWV, and JNCIA-EX, and has worked closely supporting U.S. Government agencies for the last 20 years.

Author's Acknowledgments
I would also like to thank Editor in Chief Patrick Ames for all of his hard work, guidance, and encouragement, and Copyeditor Nancy Koerbel and illustrator Karen Joice for their assistance and hard work, Richard Woodman for his technical review and for writing Appendix B, and Tim Brown for his grounding perspective and technical guidance.

ISBN: 978-1-941441-19-0 (print)
Printed in the USA by Vervante Corporation.

ISBN: 978-1-941441-20-6 (ebook)

Version History: Second Edition, August 2015
4 5 6 7 8 9 10

This book is available in a variety of formats at www.juniper.net/dayone.

What You Need to Know Before Reading This Book

Before reading this book, you should be familiar with the basic administrative functions of the Junos operating system, including the ability to work with operational commands and to read, understand, and change the Junos configuration. If you do not possess these basic competencies, the book may be harder to digest and the configuration samples more difficult to implement on your device or test bed.

If you need help polishing your Junos CLI skills, see the *Day One* suite of books at http://www.juniper.net/dayone. *Day One: Exploring the Junos CLI* and *Day One: Configuring Junos Basics* are highly recommended.

The author made a few assumptions about your networking knowledge when writing *This Week: Hardening Junos Devices, 2nd Edition*:

- You have a practical working knowledge of the TCP/IP.
- You have an intermediate-level understanding of, and configuration experience with, the Junos OS. This book expands on basic configuration concepts to enable enhanced security.
- You have a general understanding of network attacks and basic security principles.
- Although not mandatory to complete the reading of this book, it would be beneficial to have access to a Junos device on which to practice configuring the examples covered.

After Reading This Book, You'll be Able To

- Understand the non-technical aspects of device management that are critical to the overall security posture of your organization.
- Understand physical security is an important aspect of device deployment and that software features can help strengthen your devices.
- Understand the benefits of a common operating system or "One Junos," and how it streamlines device hardening.
- Understand that Junos software's inherent security features and minimalistic default configuration are the foundation for a solid security baseline.
- Identify necessary device services and harden them appropriately, while understanding the rationale for, and possible impact of, doing so.
- Identify important management, access services, and user account restrictions to provide least privileged access.
- Successfully configure route authentication for popular routing and signaling protocols.
- Correctly create and apply a firewall filter to protect the routing engine.

Before Getting Started...

Let's clarify a few topics that are frequently referenced throughout *This Week: Hardening Junos Devices, 2nd Edition*.

Security Policy

According to CERT.org, a security policy provides a framework for making specific decisions, such as which defense mechanisms to use and how to configure services. It is the basis for developing secure programming guidelines and procedures for users and system administrators alike. With a security policy in place you can create security checklists that contain lists of security practices that are specific to your organization.

Needless to say, it's a lot easier to harden your network devices if you already have a security policy, especially one that defines the minimal criteria necessary for managing and securing your devices.If you don't already have a security policy you should consider the following policy concerns prior to proceeding with this book:

- Password complexity policy: What are the minimum and maximum password lengths acceptable for your organization to consider a password as secure? A combination of numbers, upper and lower case characters, as well as special characters should also be required to meet best practices. Don't skimp on this most basic level.
- Authentication policy: Will you use local or centralized authentication? RADIUS or TACACS+?
- Access policy: What access services will be used to manage your devices (for example, SSH, J-Web?). Should encryption be required for all access services?
- Management policy: What management services do your network devices have to support (for example, NTP, SNMPv1/2/3, Syslog, SSH, etc.)?

Redundancy and Resiliency

Confidentiality, integrity, and availability are core principles of information security, with stability and predictability being the main objectives to availability. Redundant systems and a resilient design can go a long way to meet these basic concerns.

So when going through this book and hardening your devices, remember you can greatly increase reliability by always configuring things in pairs: two Syslog servers, two authentication servers, two NTP servers, etc. You can also maximize availability by ensuring the primary and backup servers are located on different network subnets, in different buildings, or in different geographical locations.

MORE? When designing a network you should design for maximum availability. There are many High Availability technologies that should be carefully considered, such as, device clustering, VRRP, Link Aggregation Groups (LAG), and JSRP, among others. For a detailed reference on High Availability, see the book, *Junos High Availability*, by James Sonderegger, Orin Blomberg, Kieran Milne & Senad Palislamovic, from O'Reilly Media, 2009, at http://www.juniper.net/books.

Juniper Knowledge Base (KB) Articles

Throughout this book there are many references to Juniper Knowledge Base (KB) articles and many KBs require an account on the Juniper Customer Support Center (CSC).

Chapter 1 discusses some of the other benefits of having an account on Juniper's Customer Support Center.

While you don't need a CSC account to read and benefit from this book, this book cannot possibly cover all the pertinent aspects of security and hardening, and so it provides these and other cross-references throughout for you to follow up with at another time.

About *This Week: Hardening Junos Devices, 2nd Edition*

Juniper takes the security of its products very seriously and has proven processes and procedures that follow industry best practices. *This Week: Hardening Junos Devices* covers these process and procedures and divides them into the following topics areas, which comprise its four main chapters:

- Non-Technical: Not all aspects of security are technical. Chapter 1 covers important security-related details about Juniper's Security Incident Response Team (SIRT) and its Customer Support Center (CSC), software downloads, vulnerability disclosure, and supply chain integrity.
- Physical Security: A malicious user with physical access to your network devices can cause damage that software features simply can't help secure. Chapter 2 covers physical access protection for your devices.
- Operating System Security: A secure base operating system and reasonable default behaviors are the foundations of the overall device security posture. Chapter 3 discusses Junos default management and kernel and network behaviors related to security.
- Configuration Hardening: Chapter 4 demonstrates the configuration of certain Junos OS features to harden the necessary aspects of the device as well as ways to preserve the hardened configuration.

In addition to these four chapters, the Appendices contain useful items that can help your security posture:

- Appendix A: A list of certifications Juniper Networks engages in to meet the U.S. Government's Approved Products List (APL).
- Appendix B: A medium-level security sample configuration is provided for Junos devices.
- Appendix C: This appendix distills the main points of the book into a handy checklist that you can use to mark off to-do items.

Welcome to *This Week*

This Week books are an outgrowth of the extremely popular *Day One* book series published by Juniper Networks Books. *Day One* books focus on providing just the right amount of information that you can execute, or absorb, in a day. *This Week* books, on the other hand, explore networking technologies and practices that in a classroom setting might take several days to absorb or complete. Both libraries are available to readers in multiple formats:

- Download a free PDF edition at http://www.juniper.net/dayone.
- Get the ebook edition for iPhones and iPads at the iTunes Store>iBooks. Search for *Juniper Networks Books*.
- Get the ebook edition for any device that runs the Kindle app (Android, Kindle, iPad, PC, or Mac) by opening your device's Kindle app and going to the Kindle Store. Search for *Juniper Networks Books*.
- Purchase the paper edition at either Vervante Corporation (www.vervante.com) or Amazon (www.amazon.com) for prices between $12-$28 U.S., depending on page length. Ship anywhere around the world.
- Note that Nook, iPad, and various Android apps can also view PDF files.

About this Second Edition

Security is always evolving and new features must be developed to keep pace with emerging threats. This Second Edition covers the new security features that Juniper has incorporated into Junos and clarifies some common questions asked from the first edition. All features discussed in this book are available and tested in Junos 12.3 (current recommended code), and it also discusses options for some features that are available in later code releases.

- Chapter 1 was revised to to reflect the changes in Juniper's security advisory process, subscribing to product notifications, and touches on Juniper's supply chain assurance and brand integrity programs.
- Chapter 2 was updated with additional physical security information regarding securing USB ports and encrypted configuration files.
- Chapter 3 now includes more information regarding password storage and protection.
- Chapter 4 includes sample firewall filter terms that allow you to custom build a filter that meets your organization's operational needs. Chapter 4 also includes a section with methods to keep your system hardened during normal operations.

About the Companion Checklist

This book includes a companion checklist that can assist you in hardening your organization's security stance as the last pages of this book. It is also available as a standalone PDF file on this book's landing page at: http://www.juniper.net/dayone or on http://www.juniper.net/posters.

Chapter 1

Non-Technical But Important

Routers, switches, and firewalls are considered critical infrastructure devices because they are the primary means of providing connectivity and security functions for your network. Just like the workstations and servers, these devices should be hardened from probes, scans, and attacks. Hardening is a systematic process of securing a device to reduce its attack surface through design, deployment, and configuration to form layers of protection. These layers of protection include the physical layer, operating system layer, protocol layer, and the user layer. When you harden a device you have to look at all aspects of that device including its physical location, networking purpose, what externally reachable services are enabled, how engineers access the device, and what privileges they should have when doing so.

Hardening a Junos device is more than just configuring firewall filters to only permit authorized connections, rate-limiting some protocols, and dropping all other traffic. This approach would be the equivalent of putting a fence around your business and not implementing any other security precautions. The fence (firewall filters) is an important component of the security, but additional steps need to be taken in case the fence is breached. A *defense-in-depth* approach should be taken to provide comprehensive security, and without taking this analogy too far, you might consider locks on doors and windows, security lighting, security guards, and in some environments fingerprint or even retina scanners.

This book doesn't tell you what security features must be implemented in your network, because different organizations will have different security requirements. Instead, it explores the various security features built into Junos and how to implement the configuration, and it then provides caveats to and the consequences of such deployment.

Ultimately, it's up to you to implement the security features that will make Junos comply with your company's security policy. Let's begin.

One Junos

The Junos Operating System provides a common language across Juniper's routing, switching, and security devices. The truly unique nature of Junos OS begins with its most fundamental virtue: a single source code base. This means that Juniper Networks developers can create new features once and then share the code, as applicable, across the many platforms running Junos OS, as shown in Figure 1.1. A single, cohesive operating system that provides a consistent user experience makes planning easier, day-to-day operations more intuitive, device security consistent, and implementing changes faster. Administrators can configure and manage functionality from the basic chassis to complex routing using the same tools across devices to monitor, manage, and update the entire network.

The majority of default behaviors are the same across Juniper's many platforms (J, M, MX, EX, SRX) and most of the features demonstrated in this book are part of the Junos core codebase – they should apply to all Junos-based platforms. Some of the commands demonstrated are specific to certain platforms and even specific to hardware modules. When the behavior is different for a specific platform it will be documented.

NOTE You'll notice that throughout this book the platform type in the configuration examples varies. It's a testament to the One Junos concept. All devices in the configuration examples were tested and verified using Junos 12.3, which at the time of this writing is still the recommended code for most Junos platforms.

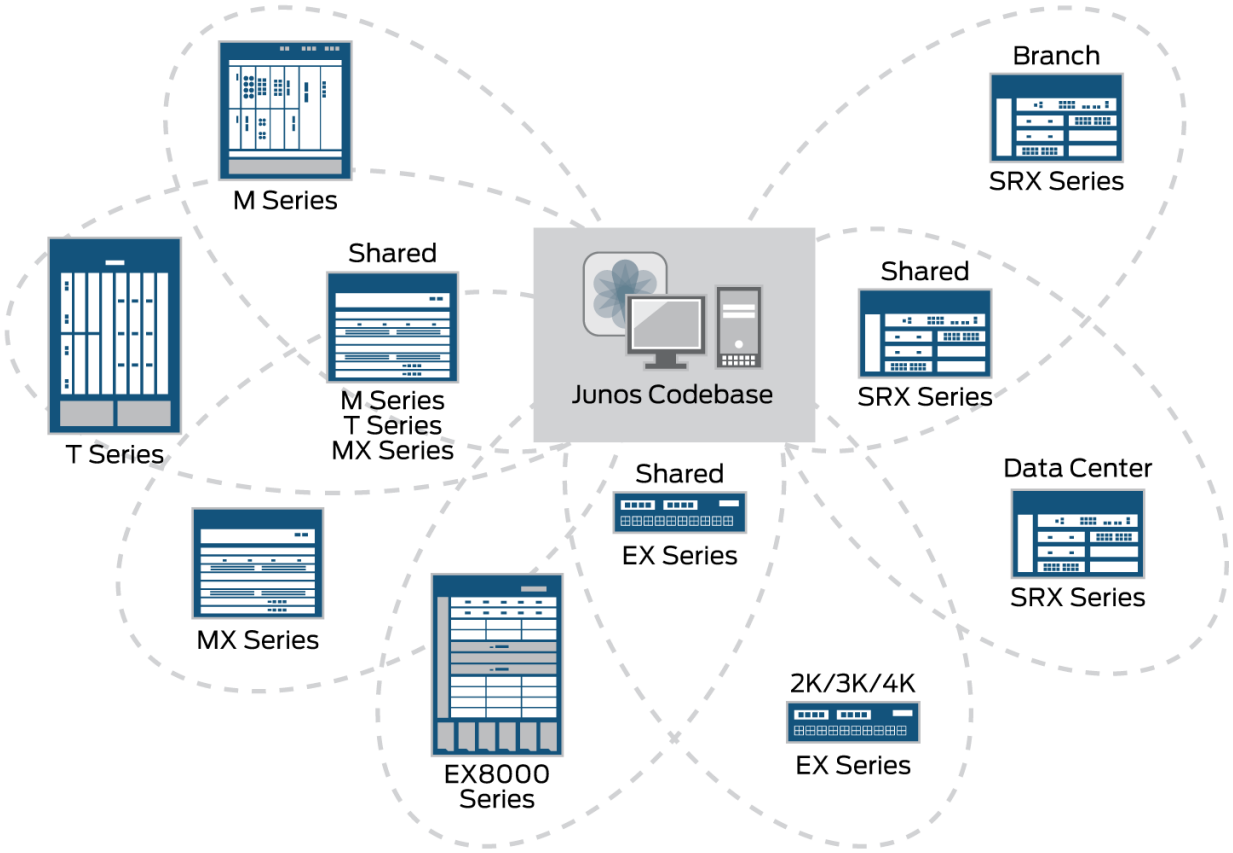


Figure 1.1 The Junos Codebase Related to Specific Platforms

Juniper Security Incident Response Team (SIRT)

The Juniper Security Incident Response Team (SIRT) is the focal point for all security vulnerabilities with, or related to, Juniper products or services. The team's role is to respond to and manage vulnerability reports from start to finish. In addition, SIRT plays a support role for customer Distributed Denial of Service (DDoS) attacks to full network penetrations.

SIRT works with the Operational Security Community, other Computer Security Incident Response Teams (CSIRT), the Juniper customer community, and other media to maintain situational awareness of the threats to Juniper products, services, or customers.

This information is processed and communicated to the larger SIRT Team, peers in the company, and ultimately to Juniper's customers. Additionally, SIRT fosters new Best Current Practices (BCPs) and drives industry operational security communities that safeguard its customers' networks.

Security Bulletin Overview

Juniper SIRT routinely publishes a couple of different types of Security Bulletins for its customers to keep them updated on the latest security related findings. The Juniper SIRT considers numerous criteria for determining if an issue warrants SIRT attention and, if so, how and to what range of products and software releases a fix will be applied and also how and when the issue will be published. Starting in 2009, SIRT Security Bulletins were divided into two groups:

- Security Advisories that describe vulnerabilities in Juniper products or services, whether specific to Junos or external sources, such as protocol flaws.
- Security Notices that discuss issues not related to a direct vulnerability in Juniper products or services, but that still deserve attention from its customers, partners, and the public (as warranted).

Both types of Security Bulletins may contain technical workarounds to assist in vulnerability mitigation.

TIP To report a suspected or confirmed vulnerability, go to: http://www.juniper.net/us/en/security/report-vulnerability/.

Entitled Disclosure

Juniper Networks used to practice Entitled Disclosure of security advisories. This meant that advisories were only available to customers and partners and not publicly announced. The intent of the entitled disclosure approach was to inform the people who needed the information while still protecting customers from people that may use the information for malicious purposes.

In 2014 Juniper Networks initiated a Public Disclosure policy, making all security policies publicly available. This change in policy reduces the chances of confusion and miscommunication of critical information by keeping Juniper Networks as the authoritative source of publicly verifiable information.

All past, present, and future Juniper Security Advisories (JSA) are now publicly accessible at http://advisory.juniper.net and JSAs are also posted to twitter at http://twitter.com/JuniperSIRT.

Advisory Release Schedule

Juniper has two types of disclosures: *Scheduled* and *Out of Cycle*. Scheduled disclosures are publicly announced quarterly on the second Wednesday of the month (January, April, July, and October) and will include advisories for all affected products.

Out of Cycle disclosures only occur if there is an active exploit of a vulnerability, an industry-wide security incident, or a third-party coordination of a multivendor issue.

This predictable release schedule helps engineers choose a secure version of Junos for their upcoming deployments – it's always easier to deploy devices with the correct code rather than having to go back and upgrade after an advisory is posted.

Vulnerability Criticality

First.org defines the Common Vulnerability Scoring System (CVSS) as: *a vendor agnostic, industry open standard designed to convey vulnerability severity and help determine urgency and priority of response. It solves the problem of multiple, incompatible vulnerability scoring systems and is intended to be usable and understandable by anyone*.

CVSS is currently used by many networking equipment vendors. It's part of the U.S. National Vulnerability Database initiative and is currently under consideration as an International Telecommunication Union standard in Study Group 17. CVSS is best thought of as a tool through which two different types of organizations can communicate vulnerability severity – reaching a common understanding of the severity so as to take appropriate action.

Juniper Networks uses CVSS version 2 to rank all our reported vulnerabilities. The CVSS Base Score is used to gauge the severity and set priorities for the fix and remediation. Starting in January 2010, SIRT began providing the CVSS Base Score in all Security Advisories, allowing customers to then use that Base Score to perform a full CVSS assessment. The total CVSS Base Score should provide Juniper's customers and users with a more precise understanding of the severity of the vulnerability as it relates to their specific network.

MORE? See Juniper's Knowledge Base Article KB16446, *Common Vulnerabilities Scoring System (CVSS) and Juniper's Security Advisories*, for more detailed information about CVSS at kb.juniper.net.

Juniper Customer Support Center (CSC)

The CSC is an award-winning customer information portal for Juniper customers and partners. The CSC provides users access to the "Problem Report" search engine, License management, Knowledge base articles, and Application notes This section of the book focuses specifically on the security-related CSC features that play an important role in the larger security picture.

MORE? You'll need to log in with your CSC account at http://www.juniper.net/customers/support/ in order to access most services.

Software Downloads

The software download section of the CSC is where engineers obtain the most recent versions of the Junos OS to install on their devices. A common concern with centralized software distribution sites is the potential risk of compromised code being uploaded and then distributed on a large scale. The CSC software download pages provide easy access to the cryptographic checksums for all Junos software so engineers can easily verify the integrity and authenticity of the download.

A cryptographic checksum is a fixed-size bit string that is calculated from a block of digital data, in this case the Junos OS install package. The checksums should then be used to validate the integrity of the download and detect accidental or intentional changes.

The two most common checksum algorithms are MD5 and SHA1. Juniper includes both MD5 and SHA1 checksums for every version of Junos OS on the CSC as shown in Figure 1.2.

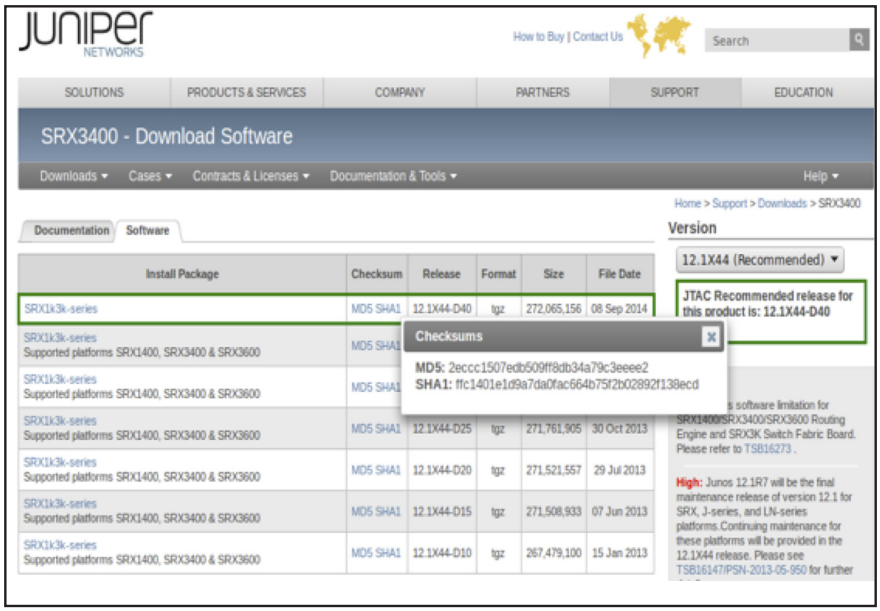


Figure 1.2 Software SHA1 Checksum

After the installation package is copied to the device, use the built-in Junos checksum utility to verify the integrity of the package prior to installation (matching checksum confirms that a genuine Juniper provided installation package was downloaded and transferred to the device unmodified):

```
=jweidley@srx3400> file checksum sha1 junos-srx1k3k-12.1X44-D40.2-domestic.tgz
SHA1 (/var/tmp/junos-srx1k3k-12.1X44-D40.2-domestic.tgz) =
ffc1401e1d9a7da0fac664b75f2b02892f138ecd
```

There are many free utilities available on the Internet for Windows, Mac, and Linux to verify the cryptographic checksums of Junos software. Freeware utilities should be carefully considered to ensure that software is obtained from a trusted source.

TIP If the checksum of the Junos install package on the device is different than the published checksum on the CSC, do not proceed. Verify the platform type and code version and download the installation package again. If the checksums are still different, open a JTAC case to resolve the problem.

MORE? See *Day One: Junos Tips, Techniques, and Templates 2011* for additional information and tips on matching checksums: http://www.juniper.net/dayone.

Technical Bulletins

Running supported Junos software on supported hardware maintains a stable network that can be easily upgraded in the event of a security vulnerability.

Juniper routinely publishes technical bulletins regarding its products to keep its customers updated with the latest information for operational and planning purposes. Technical Bulletins include End of Life (EOL) announcements for hardware and software as well as the SIRT Security Bulletins previously discussed.

CAUTION It is not a standard practice of Juniper Networks to apply security fixes to releases, which are beyond End of Engineering (EOE) or EOL. See *Juniper Networks Knowledge Base Article KB16765* for more information about which releases have had vulnerabilities fixed: kb.juniper.net.

MORE? Visit http://www.juniper.net/support/eol/ to view the EOE, EOS, and EOL dates for all hardware and software versions.

It is highly recommended that you at least sign up for the EOL and SIRT bulletins.

How to Sign Up for EOL and SIRT Bulletins

1. First go to: https://www.juniper.net/customers/support/.
2. Log in with your CSC username and password.
3. Click the *By Task* tab on the left.
4. Cick *Troubleshoot and Research*.
5. Click *Manage Technical Bulletin Subscriptions*.

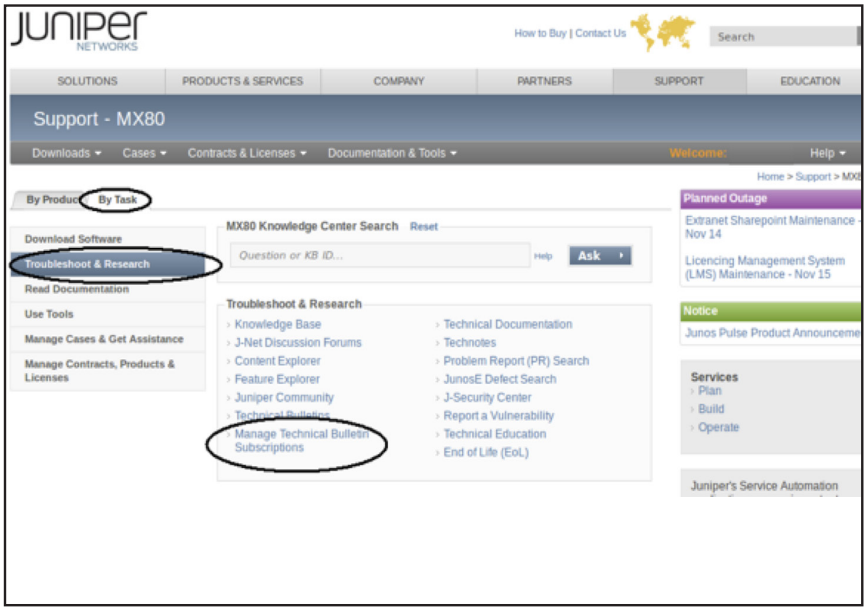


Figure 1.3 CSC Navigation to Manage Subscriptions

6. Either create a new *Security Advisories* subscription from the drop-down list or edit an existing subscription.

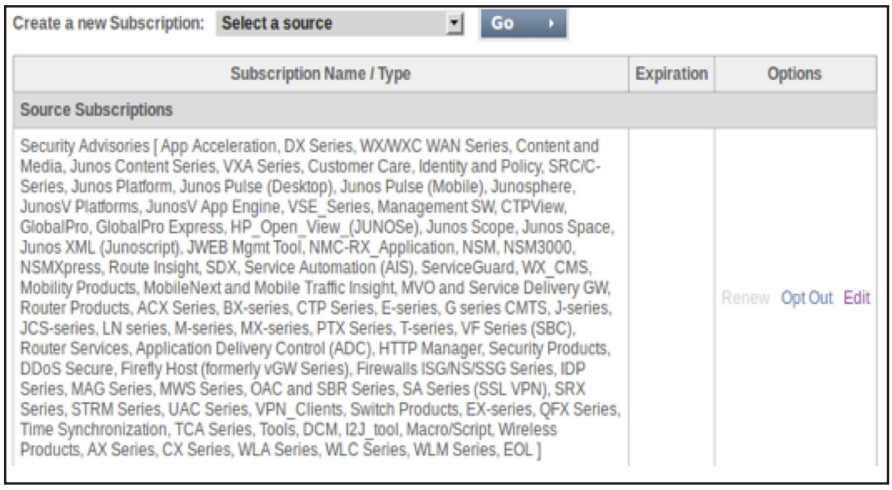


Figure 1.4 Modified View of Source Subscriptions

7. From the matrix list of products, select the products that you are interested in receiving notifications for, being sure to select EOL at the bottom of the page.

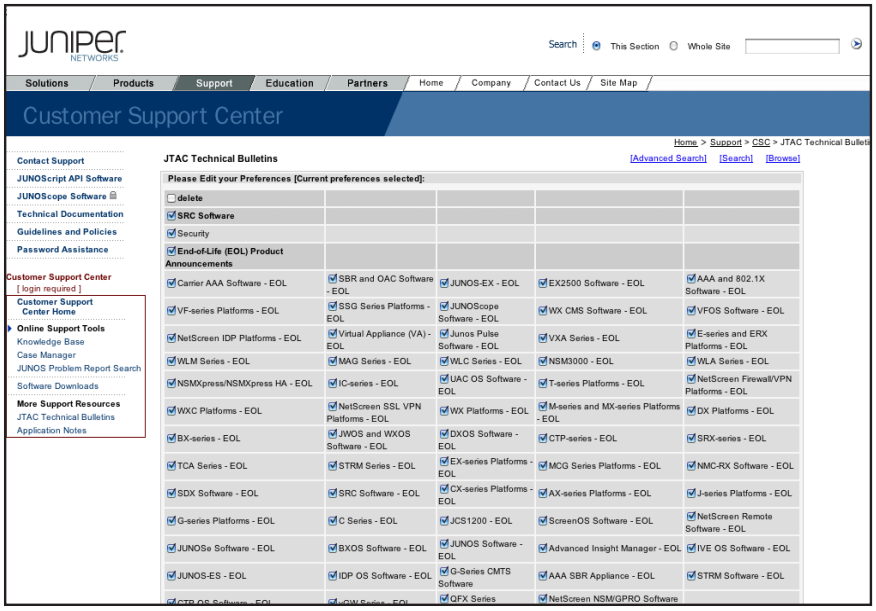


Figure 1.5 Modified View of Product list

8. Click on *Save Subscriptions*.

9. Optionally, you can repeat steps 6-8 to create a subscription for Technical Bulletins. Again, be sure you select the EOL option at the bottom of the page.

MORE? Review the *JTAC User Guide*, located on the CSC main page at http://www.juniper.net/customers/support/, for more information about other services offered via the CSC.

Supply Chain Integrity

Serious concerns have emerged in the past several years about the integrity and trustworthiness of IT products and the ability of an adversary to compromise or insert vulnerabilities into IT products through infiltration of the supply chain.

In 2001, years before supply chain integrity became a serious concern, Juniper Networks established a formal supply chain assurance and brand integrity program for securing our products and our supply chain. The Juniper brand integrity program is one component of a comprehensive corporate security plan. At Juniper, we believe brand protection programs are inherently reactive to problems discovered in the channels. Juniper's philosophy has been to implement security and integrity best practices throughout our product lifecycle process to prevent instances of counterfeit products or components, and to ensure that our customers receive the highest quality products available in the marketplace.

Juniper incorporates numerous international standards in the operation of its supply chain and brand integrity programs, including:

- ISO 27001 for information security
- ISO 9001 / TL9000 Quality management system (Certified)
- C-TPAT and AEO supply chain security criteria (Certified Tier 3 C-TPAT and AEO- Security)
- Common Criteria product certifications

Juniper also employs best practices for supply chain security from organizations such as The Open Group Trusted Technology Forum (O-TTF); the Alliance for Gray Market and Counterfeit Abatement; and the Software Assurance Forum for Excellence in Code (SAFECode). Some of these best practices include: component integrity assurance; traceability of products and components; anti-counterfeit features within our products; supplier selection (including an evaluation of foreign interests, relationships, and potential for foreign control); physical security; information and IP security; and channel monitoring and incident response. Finally, Juniper works with our industry partners and various governments to identify new and emerging risks and collaborate on best practices to mitigate those risks.

Contact your Juniper account team if you are interested in more details on Juniper's supply chain assurance and brand integrity programs.

MORE? For more information read Brad Minnis's Blog post to the Juniper Forums, http://forums.juniper.net/t5/Security-Now/Securing-the-Information-Systems-Supply-Chain/ba-p/260465 and Robert B. Dix Jr.'s statement to the Subcommittee on Communications and Technology, http://democrats.energycommerce.house.gov/sites/default/files/documents/Testimony-Dix-CAT-Cybersecurity-Supply-Chain-2013-5-21.pdf.

Chapter 2

Physical Security

There's no substitute for physical security! There hasn't been a software feature created yet that can protect a device from a determined person with a screwdriver or a hammer. So obviously it's highly recommended that critical network devices be deployed in secure locations with adequate physical security measures. Seems simple, but it's not always possible to deploy network devices in a locked rack in a secure data center or a secure wiring closet. This chapter shows you Junos features that you can implement to provide basic protection against common problems caused by someone with physical access to your Junos device.

Console and Auxiliary Ports

If not properly secured, the console and auxiliary ports can permit unauthorized access to the device. Let's begin this chapter with that premise.

The console port is used for the initial configuration of the device and for emergency access, while the auxiliary port is primarily used for remote access via a modem.

Securing the Console Port

The console port is the only access method that is enabled by default on most Junos devices, and there are two main security concerns with console port access: unattended sessions and password recovery.

This section shows you how to enable features to address these security concerns, as well as covering features to meet some of the more stringent security requirements.

CAUTION Do not consider implementing console security until you have local user accounts configured and a solid strategy of backing up your configurations to a secure remote system.

Unattended Sessions: How to Configure Basic Console Security

Here's how to implement basic console port security to address the security concerns mentioned above.

One type of unattended session occurs when an authorized engineer is physically connected to the console port and inadvertently disconnects the cable without properly logging out. The next time someone connects to the console port, they could have access to the CLI without authenticating.

The *log-out-on-disconnect* option does exactly what it says. When the console cable is physically disconnected from the device's console port that user session will be terminated. Set the option like this:

```
[edit system ports]
jweidley@ex3200# set console log-out-on-disconnect
```

The other type of unattended session occurs more commonly when terminal servers are used for remote console port access. With terminal servers the console port is always connected and engineers remotely access the terminal server and initiate access to device console ports. Some terminal servers use unique key sequences to terminate the console session and there are times when that termination doesn't always happen cleanly, leaving the user's console session still active. In this situation, another user can reattach to that console session and be logged in as the first user.

The best way to address this is to configure custom login classes with idle session

timers. This is discussed in more detail in the section on User Authentication in Chapter 4, but a short introduction is helpful at this point. Idle timers are set in login classes and every user account must have a login class defined. Junos has a few default login classes, one being super-user, which has the idle-timeout function disabled. Any user account with a login class of super-user won't be automatically logged out for inactivity whether they are on the console or accessing the device remotely.

Every Junos device has a user account named root. The root user is assigned to the super-user login class by default and that cannot be changed. To prevent the root user from logging in on the console port, set the console port as insecure. This forces users to authenticate to the console port using a regular user account (local or RADIUS/TACACS+), ideally configured with a custom login class with the idle-timeout option set.

Here's how you set the console port as insecure to prevent root user login:

```
[edit system ports]
jweidley@ex3200# set console insecure
```

Password Recovery

Juniper's public documentation regarding password recovery of Junos-based devices shows the reader how boot into single user mode to reset the root password without supplying authentication credentials. It makes sense if you've lost the root password, but it also allows someone with physical access to your device to recover the password and lock you out.

Configuring the console port with the *insecure* option protects from unauthorized password recovery by requiring the root password be entered prior to beginning the recovery process. As you can see below, when booting into single user mode, you are prompted for the root password prior to being given the recovery or shell option:

```
Enter root password, or ^D to go multi-user
Password:
Enter full pathname of shell or 'recovery' for root password recovery or RETURN for /bin/sh:
```

Now, let's review the configuration:

```
[edit system ports]
jweidley@ex3200# show
console {
   log-out-on-disconnect;
   insecure;
}
```

CAUTION If you configure the console port to `insecure` and then forget the root account password, the only way to recover the device is to perform a low level installation of the device where all configuration and log data is lost!

NOTE When the console port is configured as `insecure` and someone attempts to login as the root user, Junos will notify you of the unauthorized activity by generating a Syslog message similar to this:

```
Sep 17 05:08:11.550 2011  ex3200 login: LOGIN_REFUSED: Login of user root from host [unknown] on
device ttyu0 was refused: NOROOT
```

TIP Ensure that you configure at least one local account with maximum privileges on the system so that in emergency conditions you can still gain access to the device! See *How to Create Local User Accounts* in Chapter 4.

How to Disable Password Recovery

Sometimes any type of password recovery can be considered an unnecessary risk, such as when the Junos device is in an unsecured location, or in very high security environments.

From the previous section, you see that password recovery is still possible if you know the root password. So in order to completely disable password recovery you have to disable root user authentication to the device. This is done by using the `encrypted-password` option with a plain-text value like "`***DISABLED****`" when setting the root user's password:

```
[edit system]
jweidley@ex3200# set root-authentication encrypted-password "***DISABLED***"

[edit]
jweidley@ex3200# show system
host-name ex3200;
authentication-order radius;
root-authentication {
    encrypted-password "***DISABLED***"; ## SECRET-DATA
}
```

The root account is the most powerful user account on the system and is required for most system-level debugging. Note that the configuration above will also disable the ability to perform most advanced debugging from the shell. *Do not disable password recovery unless you're absolutely sure you need to.*

TIP The encrypted-password can be set to any plain-text value but setting it to "`***DISABLED***`" makes your configuration self-documenting and obvious even to non-Junos experts.

CAUTION If the console port is configured as `insecure` and the root account is disabled, the only way to recover the device is to do a low level installation of the device where all configuration and log data is lost!

Details: Plain-text-password versus encrypted-password

Throughout this book you will see references to the `plain-text-password` and `encrypted-password` options to various commands. It is important to understand how each of these command options function to be assured your device passwords are properly secured.

When the `plain-text-password` option is used, Junos goes through a detailed process to generate a new password hash based on the supplied password and some random data, called a *salt*. Written as an equation the process would look like this: (supplied password + salt = hash). See the example below of creating a new user account:

```
{primary:node0}[edit system login]
weidleyj@srx650# set user EMERGENCY authentication plain-text-password
New password:
Retype new password:

{primary:node0}[edit system login]
weidleyj@srx650# show
user EMERGENCY {
    authentication {
        encrypted-password "$1$.xbpsxry$Ch2shl6aFDT/UjX6JnR8v."; ## SECRET-DATA
    }
 }
```

Junos uses the encrypted-password option to store the hashed representation of the user's password and NOT the user's actual password. If someone attempted to log in using the EMERGENCY account on this device Junos would use the same (supplied password + salt =hash) process and if the result matched the `encrypted-password` value in the configuration the user would be granted access.

The **`encrypted-password`** option tells Junos not to go through the process to generate a new password hash and to accept the proceeding value as the actual password hash. In the *How to Disable Password Recovery* section above we discussed configuring the `encrypted-password` option with the plain-text value of "`***DISABLED***`". In that scenario, when someone attempts to log in the supplied password + salt would not result in a hash that matches "`***DISABLED****`" effectively making the account inaccessible.

Another benefit of using the `encrypted-password` option is that it allows a Tier 3 engineer to give a Tier 1 engineer the following commands to change the password on the emergency account without disclosing the actual password:

```
set system login user EMERGENCY authentication encrypted-password "$1$zEj/6q97$r8GfPUOfRqGhNaSa7f
Jg//"
```

For further explanation of these configuration options, check out the *Plain-text Password Versus Encrypted Password* Juniper Learning byte at https://www.youtube.com/watch?v=WdFDX13Y2Q8.

How to Disable the Console Port

It's quite possible in some extreme environments that any console access could be considered an unacceptable risk. In these cases, the console port can be completely disabled:

```
[edit system ports]
jweidley@ex3200# set console disable

[edit system ports]
jweidley@ex3200# show
console disable;
```

ALERT! Disabling the console port is not recommended. Disabling the console port will negatively impact emergency access and any recovery of the device. The console port should only be disabled when it is absolutely necessary to comply with strict physical security requirements.

TIP An alternative to disabling the console port in software is to use a physical port locking device. Several manufacturers make these devices; they are cost effective, and they provide reasonable physical security without compromising functionality.

Securing the Auxiliary Port

The auxiliary port is intended to be used with a modem to provide dial-in access to devices in remote locations. It's worth mentioning that the auxiliary port can also be used as a secondary console port, in which case it inherits some console port security concerns.

But not all Junos devices have an auxiliary port. Although the auxiliary port is disabled by default, you wouldn't know it from looking at the configuration. In this case it's a good practice to clearly document the default behavior by adding the commands to the configuration:

```
[edit system ports]
jweidley@MX240# set auxiliary disable

[edit system ports]
jweidley@MX240# show
auxiliary disable;
```

If you have a valid purpose for auxiliary port usage, the CLI also has the same `insecure` option as the console port to restrict direct root access:

```
[edit system ports]
jweidley@MX240# set auxiliary insecure

[edit system ports]
jweidley@MX240# show
auxiliary insecure;
```

Before moving on, the level of caution that must be taken when limiting access and functionality of the console and auxiliary ports cannot be stressed enough. Take the time to investigate and test every scenario, for example, day-to-day usage, maintenance procedures, emergency scenarios, and more, prior to implementing these features.

Diagnostics Ports

Some hardware modules, such as the System Control Board (SCB), System and Switch Board (SSB), Switching and Forwarding Module (SFM), and Forwarding Engine Board (FEB), have a special port that can be used for advanced diagnostics. By default, diagnostic ports are not secured by a password, which makes it possible for an unauthorized user with physical access to the device to gain access to the system and possibly obtain sensitive network specific information.

Use the commands below to password protect the two different types of diagnostics ports:

```
[edit system]
jweidley@srx650# set diag-port-authentication plain-text-password
New password: <password>
Retype new password: <password>

[edit system]
jweidley@srx650# set pic-console-authentication plain-text-password
New password: <password>
Retype new password: <password>

[edit system]
jweidley@srx650#
```

Most organizations require that static passwords be changed periodically and that would also apply to passwords on diagnostic ports. Advanced diagnostics using these ports should be an infrequent task and disabling these ports reduces administrative overhead of periodically changing the password. Junos doesn't have a configuration option to disable diagnostic ports, like it does for the console and auxiliary ports, but you can set the password to an impossible hash value like we did above in the *How to Disable Password Recovery* section. This is done using the encrypted-password option:

```
[edit system]
jweidley@srx650# set pic-console-authentication ?
Possible completions:
+ apply-groups        Groups from which to inherit configuration data
+ apply-groups-except  Don't inherit configuration data from these groups
  encrypted-password   Encrypted password string
  plain-text-password  Prompt for plain text password (autoencrypted)

[edit system]
jweidley@srx650#

[edit system]
jweidley@srx650# set diag-port-authentication encrypted-password ***DISABLED***

[edit system]
jweidley@srx650# set pic-console-authentication encrypted-password ***DISABLED***

[edit system]
jweidley@srx650# show
. . . .
diag-port-authentication {
    encrypted-password ***DISABLED***; ## SECRET-DATA
}
pic-console-authentication {
    encrypted-password ***DISABLED***; ## SECRET-DATA
}
```

MORE? See the preceding sidebar, *Plain-text Password versus Encrypted Password* for more information on the usage of the `encrypted-password` option.

NOTE These commands are specific to the hardware platform. The Junos CLI will clearly show whether the command is supported on your particular device.

Try It Yourself: Setting Diagnostic Port Password On an Unsupported Device

Log in to the device. For this example we will use an EX3200 because it doesn't have an SCB, SSB, SFM, or FEB.

Type `edit system` and press the Enter key.

Type `set ?` and look for the `diag-port-authentication` option.

Notice the command isn't present. So type the entire command `set diag-port-authentication plain-text-password` and press Enter.

Follow the password prompts to set and confirm the password.

Type `show` to view the configuration and notice the ignored configuration warning.

```
[edit system]
jweidley@ex3200# show
host-name ex3200;
##
## Warning: configuration block ignored: unsupported platform (ex3200-24t)
##
diag-port-authentication {
    encrypted-password "$1$2PbSMm6p$/0YXC//1EE7ttuxOY8LaR."; ## SECRET-DATA
}
. . .
```

TIP The Junos OS accepts and commits unsupported platform commands, even though they don't affect how the device operates. However, they are not desirable because they make your configuration slightly more difficult to read and could mislead other engineers. Consider creating a hardening template with only the required configuration options for each Junos platform type deployed in your network.

USB Ports

Most Junos devices have a USB port that can be used for auto installation, a secondary boot device, transferring files, or even a modem for remote access. But if the device is deployed in an unsecure environment, it might be wise to disable the USB port to eliminate the possibility of unexpected events.

TThe SRX Series Services Gateway and J Series Services Router each have a feature that allows the administrator to disable the USB ports in order to block users from connecting a USB mass storage device. After disabling the USB port, if a USB device was already mounted and connected, this feature unmounts and disables the device. Also any transactions in progress on the USB device are aborted.

Use the command below to disable the USB port:

```
[edit chassis]
jweidley@srx210-voodoo# show
usb {
   storage {
      disable;
   }
}

[edit chassis]
jweidley@srx210-voodoo#
```

NOTE Some Junos platforms, typically branch SRX, allow the USB system process to be disabled by using the `set system processes usb-control disable` command. Check the platform-specific Junos documentation to verify that this is a supported command.

Craft Interface and LCD Menu

Most Junos device platforms have either a craft interface or an LCD that can be used for viewing system status and alarms. There are also physical buttons or menu options to perform system control and maintenance functions, like bringing an FPC offline or online, restoring the factory default configuration, etc.

This section defines some of the features provided by the craft interfaces and LCD menus, and the options that are available to harden the device from someone with physical access.

Securing the Craft Interface

All high-end Junos platforms, except EX series, have always had a craft interface. There may be a scenario, such as a shared rack with a partner organization, that may make you consider further securing the craft interface. Disabling the craft interface

will disable the ability for someone to online/offline and FPC or PIC, acknowledge alarms, etc.

```
[edit]
jweidley@MX240# set chassis craft-lockout
```

CAUTION Consider your maintenance and support model prior to implementing this feature – it may impact an engineer's ability to perform routine maintenance on the system.

How to Secure the Reset Config Button

Smaller form factor devices, like the J Series or branch SRX, don't have a craft interface or an LCD menu, per se, but do have a *Reset Config* button that has a couple of useful options related to physical security.

The Reset Config button is located near the power button, as show in Figure 2.1, and is recessed to prevent it from being accidently pressed. In order to activate the Reset Config button it is best to use a small object, such as a straightened paper clip.

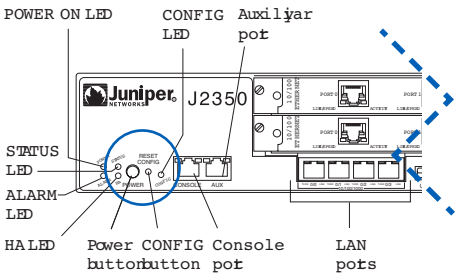


Figure 2.1 Reset Config Button Location

The default behavior of the Reset Config button is:

- Pressing and quickly releasing the button loads and commits the rescue configuration, provided that the rescue configuration has previously been saved.
- Pressing and holding the button until the STATUS LED blinks red (~ 15 seconds), deletes all configurations on the device, including the backup and rescue configurations, and loads and commits the factory default configuration.

If you want to prevent the Reset Config button from setting the router to the factory default configuration, but still want it to be able to restore the rescue configuration, use the following command:

```
[edit chassis]
jweidley@j6350# set config-button no-clear
```

If you want to prevent the Reset Config button from setting the router to the rescue configuration but still want to allow the router to be reset to factory default, use the following command:

```
[edit chassis]
jweidley@j6350# set config-button no-rescue
```

If neither of these options fits your environment and you wish to completely disable the Reset Config button, use the following command:

```
[edit chassis]
jweidley@j6350# set config-button no-clear no-rescue
```

MORE? For more information, see http://www.juniper.net/techpubs/en_US/junos14.1/topics/reference/configuration-statement/config-button-edit-chassis.html.

Securing the LCD Menu

Some devices have an LCD and menu buttons that provide convenient access to status and maintenance functions. This section will explore the available options so you are able to properly secure the LCD for your environment.

Junos OS version 10.2 introduced a new command syntax, as well as a certain granularity, to what is displayed and which options are available.

To see which menu options are available by default, use the following:

```
jweidley@ex3200> show chassis lcd menu
status-menu
status-menu power-status
status-menu environ-menu
status-menu show-version
maintenance-menu
maintenance-menu halt-menu
maintenance-menu system-reboot
maintenance-menu rescue-config
maintenance-menu factory-default
```

To show how to disable certain menu options, let's explore the scenario where this switch is located at a remote office where there are Tier 1 technicians who can assist you with simple tasks. In this example we will disable their ability to see which version of Junos is installed and the ability to overwrite the active configuration with the factory default configuration:

```
[edit]
jweidley@ex3200# edit chassis lcd-menu

[edit chassis lcd-menu]
jweidley@ex3200# set fpc 0 menu-item "status-menu show-version" disable

[edit chassis lcd-menu]
jweidley@ex3200# set fpc 0 menu-item "maintenance-menu factory-default" disable

[edit chassis lcd-menu]
jweidley@ex3200# commit and-quit comment "disabled LCD ver & load default"
commit complete
Exiting configuration mode

jweidley@ex3200> show chassis lcd menu
status-menu
status-menu power-status
status-menu environ-menu
maintenance-menu
maintenance-menu halt-menu
maintenance-menu system-reboot
maintenance-menu rescue-config
```

If the device is in an unsecure location and none of these menu options are suitable, use the following commands to completely disable the LCD menus and verify the changes:

```
[edit]
jweidley@ex3200# edit chassis lcd-menu

[edit chassis lcd-menu]
jweidley@ex3200# set fpc 0 menu-item maintenance-menu disable

[edit chassis lcd-menu]
jweidley@ex3200# set fpc 0 menu-item status-menu disable

[edit chassis lcd-menu]
jweidley@ex3200# commit and-quit comment "disabled LCD menus"
commit complete
Exiting configuration mode

jweidley@ex3200> show chassis lcd menu

jweidley@ex3200>
```

NOTE The command syntax varies slightly on the EX platforms mostly due to its virtual chassis capabilities. Notice in the above examples that the commands are specific to the FPC. This means that there will be additional configuration required for each switch in a virtual chassis.

MORE? For a detailed listing of menu items and specific platform support visit http://www.juniper.net/techpubs/en_US/junos14.1/topics/reference/configuration-statement/menu-item-edit-chassis.html.

Unused Network Ports

A recommended security practice is disabling unused network ports, or at least configuring them to be in an unusable or harmless state. This practice reduces the chances of someone getting network access to other devices or other information, while having physical access to the device.

An interface can be disabled by using the disable option, and adding an interface description is very helpful:

```
[edit interfaces]
jweidley@MX80# set ge-1/0/1 description "---unused---" disable

[edit interfaces]
jweidley@MX80# commit and-quit comment "disabled ge-1/0/1"
commit complete
Exiting configuration mode

jweidley@MX80> show interfaces descriptions
Interface   Admin Link Description
ge-1/0/0    up    up   ex4500-201;ge-0/0/0;192.168.46.2
ge-1/0/1    down  down ---unused---
ge-1/0/2    up    up   srx5600;ge-0/0/0;192.168.46.54
fxp0        up    up   Out-of-band mgt net
```

Note that EX switches have a secure default behavior related to unused ports:

- By default all ports are defined as access ports and placed in an untagged VLAN named `default`.
- Ports default to access mode and will not automatically try to negotiate trunk behavior if connected to another switch.
- Also the default trunking behavior is to not include the `default VLAN`. Notice from the output below that the `default VLAN` does not have an 802.1q tag assigned:

```
jweidley@ex3200> show vlans default
Name          Tag    Interfaces
default
                     ge-0/0/5.0, ge-0/0/6.0, ge-0/0/9.0,
                     ge-0/0/18.0, ge-0/0/19.0
dmz           200
                     ge-0/0/7.0*, ge-0/0/8.0*
```

MORE? For a detailed discussion regarding Junos switching default and native VLANs behavior, see the book, *Junos Enterprise Switching* (by Reynolds & Marschke, O'Reilly Media, 2009) at http://www.juniper.net/books.

Protecting Configuration Data

Device configurations contain sensitive information such as IP addresses, account names, passwords and protocol authentication keys, and traffic filtering security policies. You should take steps to ensure your site's proprietary data is protected when devices are deployed in unsecured locations, stored in the warehouse, and sent back to Juniper for product replacement (RMA).

Encrypting Configuration Files

By default, Junos devices store configuration files in unencrypted format on both internal and external media. This storage method could be considered a security risk because the storage media card can be removed from the device. To prevent unauthorized users from viewing sensitive information in configuration files, you can encrypt them.

If your device runs the Canadian and U.S. version of Junos software, the configuration files can be encrypted with the Advanced Encryption Standard (AES) or Data Encryption Standard (DES) encryption algorithms. If your device runs the international version of JUNOS Software, the files can be encrypted only with DES.

The encryption key used to encrypt the configuration files is stored in the device's EEPROM and not in the configuration file. If all of the devices in your environment share the same encryption key, you can copy the encrypted configuration files from one device and decrypt them on a different device. To prevent encrypted configuration files from being copied to another device and decrypted, you can use the `unique` option which incorporates the device's chassis serial number as part of the encryption key.

In this example we'll configure our Junos device to use encrypted configurations with AES encryption:

1. Set the encryption key that is used to encrypt your configuration files. Ensure you choose a complex password that complies with your organization's password policy:

```
jweidley@srx210> request system set-encryption-key algorithm aes
Enter EEPROM encryption key:
Verify EEPROM encryption key:
*** Warning: Configuration files not encrypted because
***        /config/juniper.conf was not encrypted
*** Warning: Use 'set system encrypt-configuration-files' and 'commit'
***        to encrypt JUNOS configuration files in /config
```

2. Enter the configuration mode, using the `configure` command and follow the instructions from the command output and configure the device to encrypt configuration files:

```
jweidley@srx210> configure
Entering configuration mode
The configuration has been changed but not committed

[edit]
jweidley@srx210# set system encrypt-configuration-files
```

3. Commit the configuration change with a helpful comment:

```
[edit]
jweidley@srx210# commit and-quit comment "enabled encrypted configuration files"
commit complete
Exiting configuration mode
```

The encryption process encrypts only the configuration files in the `/config` and `/var/db/config` directories. Files in subdirectories under these directories are not encrypted. The filenames of encrypted configuration files have the extension .gz.jc. Let's take a look at the `/config` directory after encrypted configuration files have been implemented:

```
jweidley@srx210> file list detail /config/

/config/:
total blocks: 60
drwxrwxr-x  2 root  wheel      512 Sep 3   2010 .snap/
-rw-r-----  1 root  wheel     1409 Feb 23 22:32 juniper.conf.1.gz
-rw-r-----  1 root  wheel     1405 Feb 23 21:52 juniper.conf.2.gz
-rw-r-----  1 root  wheel     1425 Feb 24 20:27 juniper.conf.gz.jc
---x--x---  1 root  wheel       32 Feb 23 22:51 juniper.conf.md5*
drwxr-xr-x  2 root  wheel      512 Feb 23 21:58 license/
-rw-r--r--  1 root  wheel        0  Mar 22  2013 license-status.db
-rw-r-----  1 root  wheel     1369 Feb 23 01:58 rescue.conf.gz
-rw-r--r--  1 root  wheel     1512 Feb 18 00:42 usage.db
```

There are a few important things to point out from the output above:

- The active configuration (`juniper.conf.gz.jc`) has been encrypted but the rollback files have not. To keep all of the rollback configurations protected, implementing encrypted configurations should be one of the first things that you do when configuring a new device or the unencrypted rollback configuration files should be deleted prior to deployment.
- The rescue configuration (`rescue.conf.gz`) is still unencrypted. The rescue configuration can be protected by running the `request system configuration rescue save` command.

MORE? For more information see the modifying encryption keys and encrypting and decrypting configuration files sections of the *Junos OS Library for Security Devices*, https://www.juniper.net/techpubs/en_US/junos12.1x46/information-products/pathway-pages/security/security-swconfig-initial-device-config.html#administration.

NOTE This functionality is not available or supported on all Junos platforms. Be sure you research your specific platforms, prior to implementation, to ensure supportability and proper operation.

Wiping Configuration Data

When it is necessary to remove all customer-specific data from a Junos device, there are a couple of options to choose from:

- Manually deleting important data: this is the most error prone and time-consuming solution because Junos software stores data in various places and it's hard to make sure all sensitive information is identified and deleted.
- Reformatting the Routing Engine media: this option reuses the well-defined recovery installation process as a security feature – mainly the storage media is reformatted and repartitioned, and there's a complete reinstallation of the Junos OS, with all information being lost.
- Physically remove all storage media: this is the most expensive option and requires prior coordination with your Juniper account team to ensure the security uplift package is utilized. The account team can also provide a letter of volatility that identifies all volatile and non-volatile media for all products.

NOTE Notice that the `load factory-default` configuration command is not listed above, that's because it is not intended to be a method to securely remove sensitive customer data. It merely provides a way to quickly return the configuration of the device to a clean default configuration.

This section focuses on the second option: reformatting the Routing Engine media. Note that the procedures for performing a media install are different for various Junos platforms, so this section provides general information on the process and reference links on how to actually perform the procedure.

First, it's important to understand the types of Junos OS packages that are available:

- Installation Bundle: The installation bundle can be used to downgrade or upgrade the Junos OS between minor revisions (from Release 9.1 to Release 9.2, for example). When used, the installation bundle modifies only the files required for the upgrade or downgrade between versions.
- Installation Package: The installation package is used to upgrade and downgrade from one major release to another (from Release 9.2 to Release 10.1). When installed, the installation package completely reinstalls the software, rebuilds the Junos file system, and may erase system logs and other auxiliary information from the previous installation. The installation package does, however, retain the configuration files from the previous installation.

- Installation Media: The installation media is used to recover a router from a software failure. The installation media repartitions the media and completely reinstalls the Junos OS. No information from previous installations is retained during this installation.

Installation media is the software distribution required to create a Junos disk image on a PCMCIA, Compact Flash, or USB drive that can then be used to reformat the Routing Engine and perform a media installation.

TIP Although performing a media install provides reasonable assurance that your company's proprietary data will be wiped from the Routing Engine, you should be aware that a comprehensive forensic analysis might still be able to recover your data. For high security environments physical removal of media may be the only acceptable option.

Obtaining the Installation Media

Installation media can be obtained from the Juniper software download site in the same section where Junos installation packages are located as shown in Figure 2.2.

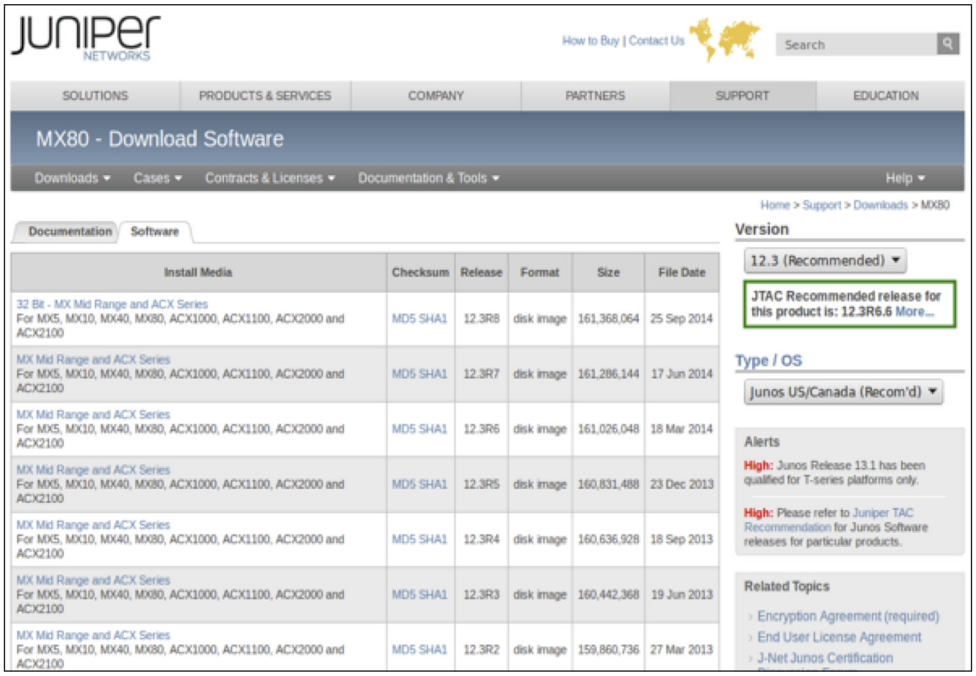


Figure 2.2 CSC: Install Media Download

There are different installation media for different Junos device platforms (i.e. 32bit, 64bit, etc.), so pay careful attention to downloading the correct package for your device(s).

MORE? For more information regarding the different Junos OS media, or creating an emergency recovery disk and performing a recovery installation, see the *Junos OS Software Installation and Upgrade Guide* at http://www.juniper.net/techpubs/en_US/junos14.1/information-products/pathway-pages/software-installation-and-upgrade/software-installation-and-upgrade.pdf.

TIP There is a `request system zeroize media` Junos CLI command that securely erases all data on the Routing Engine and resets all key values – but its functionality and support varies on different Junos device platforms. Ensure that you research the expected behavior and support status on your platforms prior to using it.

Chapter 3

Operating System Security

The Junos OS was first introduced in Service Provider environments where stability and security are critical and outages can impact hundreds of thousands of users. Service Provider environments need rational and predictable behavior in every feature. For instance, consider the directed broadcast and IP Source routing behaviors. These are sometimes useful features but it is not necessary, rational, or secure to have them enabled by default. They should only be implemented as needed and only on the specific interfaces that require those features.

The inherent security of the operating system is the foundation of the overall security and stability of the device. If the operating system isn't secure the device can sometimes be compromised regardless of the configuration.

Juniper has many inherent security features built into the Junos OS codebase that provide increased security and stability – namely, secure default values, rational network feature behavior, and kernel protection. This chapter covers those inherent security features in Junos and the default configuration that contributes to the overall security of the device.

Factory Default Configuration

Unfortunately, every listening service and enabled feaure on a network device exposes it to some level of risk. Juniper doesn't attempt to predict which services you might want enabled on your Junos device because every network is unique.

Instead, Juniper provides a very minimalistic default configuration and requires engineers to configure the features they need. This may result in more configurations prior to the initial deployment, but the security benefits far outweigh the minor inconvenience.

The factory default configuration varies a bit from platform to platform, but remember it includes only the essential configuration required for the device to function in its network role.

Try it Yourself: Viewing the Factory Default Configuration

The factory default configuration will be different depending on which Junos device you're looking at, but the key things to look for are the default accounts, access services, and enabled features:

1. Log in to your Junos device and enter into configuration mode.

2. Enter the `load factory-default` command:

```
[edit]
jweidley@MX80# load factory-default
warning: activating factory configuration

[edit]
jweidley@MX80#
```

3. Enter the `show` command to view the factory default configuration:

```
[edit]
jweidley@MX80# show
## Last changed: 2011-03-14 08:25:13 UTC
system {
   syslog {
      user * {
```

```
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
        file interactive-commands {
            interactive-commands any;
        }
    }
    ## Warning: missing mandatory statement(s): 'root-authentication'
}

[edit]
jweidley@MX80#
```

Notice that there aren't any access services (for example telnet, SSH) or accounts as part of the factory default configuration. Also, take note of the warning that the root user does not have a password – this will be discussed in the next section.

4. Enter the `rollback` command to synchronize the active and candidate configurations, in case you don't want to commit these changes:

```
[edit]
jweidley@MX80# rollback
load complete

[edit]
jweidley@MX80#

5. Verify that no configuration changes will be made:

[edit]
jweidley@MX80# show | compare

[edit]
jweidley@MX80#

6. Exit the device.
```

No Hidden Support Accounts

Some vendors of UNIX-based operating systems include one or more support accounts with root level privileges to assist customers with troubleshooting. It's commonly done by creating an account and setting the user ID (UID) or group ID (GID) to 0 in `/etc/passwd`. This should raise major security concerns because no one outside your organization should have login names and passwords for your devices. The Junos OS does not include any hidden or backdoor support accounts. You can verify that there is only one root level account by running the command below from the shell:

```
jweidley@ex3200> start shell
% cat /etc/passwd | egrep ':0:'
root:*:0:0:Charlie &:/root:/bin/csh
%
```

You can see in the output that only one account that has a UID (3rd field) and GID (4th field) set to 0, and that's the root account.

Passwords

Problems with passwords, whether it's default passwords, weak passwords, password storage, or access to passwords remains a common problem with IT devices. This section will address how Junos handles the common password security concerns.

No Default root Account Password

The root account is the most powerful account on your Junos device. It needs to be protected. All Junos devices ship without a default root password, eliminating the possibility of the default password not being changed by engineers prior to device deployment.

The root password must be configured during the initial configuration of your device. In fact, in the normal operation of the Junos OS your initial configuration will not commit unless a root password is configured:

```
[edit]
jweidley@mx80# commit
[edit]
  'system'
    Missing mandatory statement: 'root-authentication' error: commit failed: (missing statements)

[edit]
jweidley@mx80#
```

Default Password Policy

To reduce the likelihood of easily guessable passwords Junos has the following default requirements for plain-text passwords:

- Length: The password must be between 6 and 128 characters long.
- Characters: You can include most character classes in a password (uppercase letters, lowercase letters, numbers, punctuation marks, and other special characters). Control characters are not recommended.
- Complexity: Valid passwords must contain at least one change of case or character class.

The default policy provides basic protection from overly simple passwords. If this default password policy does not meet your password complexity requirements, see the section on *User Authentication* in Chapter 4 to see how to customize these parameters to create a password policy to meet your requirements.

Password Storage

The Junos OS supports several methods for securing user passwords using several different hashing algorithms. The default hashing algorithm is MD5 but that can be changed to SHA for high security environments. In this section we will discuss how Junos stores various types of authentication data.

Device specific authentication information uses a salted hash. A salt is a fixed length randomly generated string of data. The salt is unique and not reused, making brute force guessing attempts exponentially more difficult. When these passwords are set, Junos takes the user-supplied password from the CLI, incorporates the salt prior to generating the hash (supplied password + salt = hash) and stores it in the configuration. The hash that is stored in the configuration is a cryptographic representation of

the user supplied password and does not contain the actual password. These passwords are secure and can only be recovered using time-consuming and resource intensive brute-force password cracking techniques.

With device specific authentication data a user is authenticating to the Junos device and the user-supplied password is hashed and compared to the hashed representation of the user's password in the configuration file. But there are features that require Junos to supply a password in order to authenticate itself with another device or service (therefore route authentication, NTP, etc.). For these situations the passwords must be stored in the configuration file and stored in a less secure manner than the salted hash method described above. These authentication data types are referred to as *$9 passwords* and utilize a reversible encryption scheme that can be decoded with the appropriate publicly available tools. $9 passwords are stored in the Junos configuration in encoded form to protect from casual visual inspection. These encoded passwords are one of the main reasons why restricting visual access to stored passwords is critical, which is covered in the next section.

Password Access

The Junos OS identifies user passwords, authentication keys, shared secret passwords, SSH keys, and certificates in the configuration with the "## SECRET-DATA" label. This is to notify engineers that this is sensitive information that should be protected. The SECRET-DATA label also makes this information easy to filter. If you had to provide your Junos configurations to someone you could use the `show configuration | except SECRET-DATA` command to filter out sensitive authentication data.

Access to sensitive authentication data should be restricted to your most trusted engineers and limited access requires different login classes with the appropriate permissions. The section *How to Create Custom Login Classes* in Chapter 4 provides sample login classes for your consideration.

MORE? Also see *Try It Yourself: Protecting ## SECRET-DATA* in Chapter 4 for sample command output from different login classes to see how authentication data is displayed based on permissions.

Another important aspect related to password access is securing configuration backups. Ensure configurations are transferred using an encrypted protocol, such as SSH, to eliminate the possibly of password being viewed/snooped in transit. See the *Configuration Backups* section in Chapter 4. The archive server that stores the configuration files should use whole disk encryption and user access and permissions should be restricted.

MORE? For more information regarding special requirements for Junos plain-text passwords see: http://www.juniper.net/techpubs/en_US/junos14.1/topics/concept/authentication-plain-text-password-requirements.html.

SNMP Set Operation is Not Supported

When the Junos OS was first introduced, SNMP version 1 and 2 were the only versions available and both were considered insecure because the communications between the SNMP manager and the device were neither encrypted nor authenticated. The concern was that someone could recover the community string by capturing traffic and then use it to make unauthorized changes to a device.

It is not possible to configure a Junos device via SNMP because there are very few writable Management Information Bases (MIBs). Although SNMPv3 adds both authentication and encryption to the protocol, and Junos supports SNMPv3, SNMP "sets" are still not permitted.

There are many powerful ways to configure a Junos device, such as the CLI, via J-Web, or using Junoscript, but SNMP is not one of them.

NOTE There are a few writable MIBs that are used for troubleshooting, specifically the Ping and Traceroute MIB branches.

No Default Remote Access

Every enabled service on a network device increases its attack surface by supplying potentially malicious users with more opportunities to penetrate the device.

The factory default configuration for most Junos devices does not include any remote access services such as telnet, SSH, or J-Web. This secure default configuration eliminates attack vectors by forcing engineers to enable only the remote access services that are required.

NOTE Lower-end platforms, like the branch SRX and the J-Series devices, are the exception. The default configuration of these platforms does include some management services as a convenience for less experienced engineers. To reduce security concerns on these devices, management services are enabled on specific interfaces or security zones, not globally. Review the platform-specific documentation for details.

MORE? See Chapter 4's *Access Security* section for more information on disabling insecure access services and how to harden secure access services.

Kernel

It is a well known fact that Junos was based upon FreeBSD and it is possible to get access to the Unix shell from the Junos CLI. The perceived threat here is that programs could be compiled on another system and then executed on a Junos device. Juniper has anticipated this attack vector and introduced a feature that will only allow official Juniper compiled programs to be installed and executed on the Routing Engine.

Software Integrity Verification

As discussed in Chapter 1, all Junos images have publicly available cryptographic checksums that should be used to verify the integrity of the package prior to installation. Inside the installation bundle are individual packages and each of these packages have cryptographic checksums that are verified prior to installation. You can see evidence of this during the Junos install:

```
Verified jinstall-ex-4200-12.3R9.4-domestic.tgz signed by PackageProduction_12_3_0
Checking package integrity...
Running requirements check first for jbundle-ex-4200-12.3R9.4-...
Running pre-install for jbundle-ex-4200-12.3R9.4-...
Installing jbundle-ex-4200-12.3R9.4- in /tmp/installer/jbundle-ex-4200-12.3R9.4-domestic
```

```
Running post-install for jbundle-ex-4200-12.3R9.4-...
Verified SHA1 checksum of fips-mode-powerpc-12.3R9.4.tgz
Verified SHA1 checksum of jbase-ex-12.3R9.4.tgz
Verified SHA1 checksum of jboot-ex-12.3R9.4.tgz
Verified SHA1 checksum of jcrypto-ex-12.3R9.4.tgz
Verified SHA1 checksum of jdocs-ex-12.3R9.4.tgz
Verified SHA1 checksum of jkernel-ex-12.3R9.4.tgz
Verified SHA1 checksum of jpfe-ex42x-12.3R9.4.tgz
Verified SHA1 checksum of jroute-ex-12.3R9.4.tgz
Verified SHA1 checksum of jswitch-ex-12.3R9.4.tgz
Verified SHA1 checksum of jweb-ex-12.3R9.4.tgz
```

Software verification isn't just performed during installation, it also occurs during the normal operation of the device. Juniper Networks platforms, with Junos 7.5 and later, run only programs supplied by Juniper Networks. Each Junos software image includes a digitally signed *manifest* of authorized Juniper files and their correct cryptographic checksum. During the boot up process, those checksums are validated and if successful those executables are registered with the system. During normal operation of the device, Junos software will not run a program if it hasn't been previously registered with the system. This feature protects the system against unauthorized software and activity that might compromise the integrity of your device.

You can see evidence of this feature when the device is booting up:

```
Verified manifest signed by PackageProduction_12_3_0
Verified jboot signed by PackageProduction_12_3_0
Verified jbase-ex-12.3R9.4 signed by PackageProduction_12_3_0
Mounted fips-mode-powerpc package on /dev/md2...
Verified manifest signed by PackageProduction_12_3_0
Verified fips-mode-powerpc-12.3R9.4 signed by PackageProduction_12_3_0
Mounted jcrypto-ex package on /dev/md4...
Verified manifest signed by PackageProduction_12_3_0
Verified jcrypto-ex-12.3R9.4 signed by PackageProduction_12_3_0
Mounted jdocs-ex package on /dev/md6...
Verified manifest signed by PackageProduction_12_3_0
Verified jdocs-ex-12.3R9.4 signed by PackageProduction_12_3_0
Mounted jkernel-ex package on /dev/md8...
Verified manifest signed by PackageProduction_12_3_0
Verified jkernel-ex-12.3R9.4 signed by PackageProduction_12_3_0
```

TIP There is a `show system audit` command that will show the permissions, size, and checksum for all system files on your Junos device. You can use the output of this command to run periodic integrity checks on the device.

MORE? See Juniper Network Knowledge Base Article KB12831, *Verification of Junos Software Images*, for additional details (you'll need to log in with your Juniper account).

Network

Historically there have been nonessential networking capabilities that have caused instability and performance degradation on the Internet and in enterprise networks. This section discusses some of the inherent behavior of the Junos OS with reference to these capabilities, and discusses the benefits of out-of-band management.

Out-of-Band Management Interfaces

There are essentially two ways to manage your network devices: in-band or out-of-band. With in-band management, management traffic shares the same physical interfaces and cables with user traffic. With out-of-band management, separate interfaces and links are provisioned specifically for management traffic. Out-of-band management is the more expensive option but it provides clean separation between user and management traffic as shown in Figure 3.1.

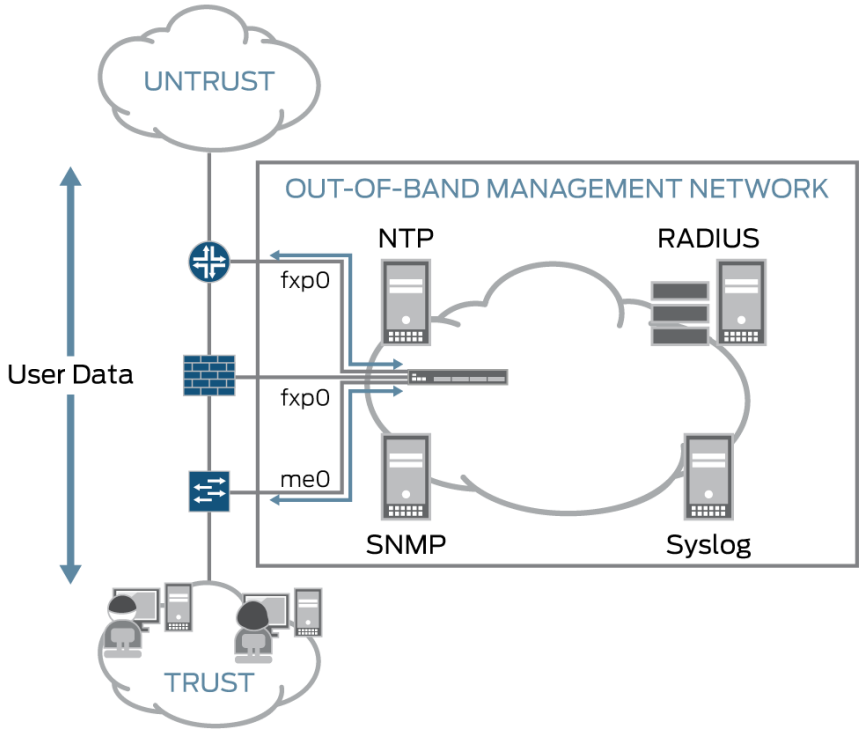


Figure 3.1 Out-of-Band Management Network

Most Junos OS platforms have a built-in, dedicated out-of-band management interface. It's called *fxp0* on high-end routing and security platforms, and *me0* or *vme* on switching platforms. Due to their smaller form factor, J-Series routers and branch SRX devices do not have a dedicated management interface (but any of the built-in Ethernet interfaces can be used for this purpose).

The dedicated out-of-band management ports are referred to as a non-transit interface, meaning that traffic that enters the device from these interfaces can not be routed out regular interfaces. The same is true for traffic that enters the router from a regular interface; it cannot be routed out the dedicated management port.

Figure 3.1 shows how using the out-of-band management ports increases the overall security posture by physically separating management data (NTP, SNMP, Authentication, Syslog, etc.) from user traffic. Malicious users can not sniff, or attempt to disrupt, management traffic if they're physically separated from it.

IP Source Routing

Routers communicate using routing protocols and exchange information about the networks they know how to reach. Packets are routed through a network based on the destination address contained in the packet. This is reasonable behavior because routers are supposed to know the best way to forward packets to the destination.

The original IP RFC (RFC 791) included IP header options for strict and loose source routing. With source routing the sender of a packet can specify either the exact path (strict) or a single gateway (loose) that packets must traverse in order to get to the destination.

The intent of these options was to assist in network troubleshooting but malicious users can also use source routing to direct packets to specific network segments, gather network topology information, and possibly subvert security restrictions.

In older versions of Junos, source routing was enabled by default, but could be disabled using this command:

```
[edit]
jweidley@MX240# set chassis no-source-route
```

Starting with Junos 8.5 and later, IPv4 source routing is disabled by default. If source routing is required in your network, it can be enabled under the `[edit routing-options source-routing]` hierarchy. Since there was a change in behavior, the Junos CLI will flag the preceding CLI command as deprecated in newer releases of code:

```
[edit]
jweidley@mx240# show chassis
no-source-route; ## Warning: 'source-route' is deprecated
```

IP Directed Broadcast

A directed broadcast occurs when a packet is sent to the broadcast address of a subnet. IP directed broadcast packets traverse the network in the same way as unicast IP packets until they reach the destination subnet. When they reach the destination subnet, and if IP directed broadcast is enabled on the last device in the path, the router or switch translates (explodes) the IP directed broadcast packet into a broadcast that floods the packet on the target subnet. All hosts on the target subnet receive the IP directed broadcast packet. This does put an unnecessary burden on the router, or switch, and directed broadcasts can be even more lethal when IP spoofing is used. This can be a recipe for a Denial of Service (DoS) attack of a particular host or to degrade the overall performance of the network.

Directed broadcasts aren't always malicious. Some remote administration tasks such as backups and Wake on LAN (WOL) need IP directed broadcast in order to function correctly.

Junos software has directed broadcasts disabled by default. Recognizing that there are valid purposes for directed broadcasts, Juniper introduced a feature called *targeted broadcast* that must be deliberately enabled to provide the functionality on a per interface basis.

Proxy ARP

Proxy ARP (Address Resolution Protocol) is a technique where a networked device, usually the router, will answer an ARP request for another device. At that point the router is responsible for routing the packet to the destination.

Proxy ARP could potentially be used by a malicious user in an attempt to put unnecessary load on the Routing Engine. Another side effect of proxy ARP is that it masks configuration errors, such as incorrect subnet masks that can cause network outages.

By default Junos software does not participate in proxy ARP and will only answer ARP requests for its own IP addresses. This behavior keeps your network configuration clean and predictable.

Proxy ARP is a great tool to have when you need it. The Junos OS does allow proxy ARP to be configured per interface as needed, but it should be considered a short term solution.

Default ARP Policer

With the exception of the Juniper EX Ethernet Switch platforms, by default the Junos OS has a built-in rate-limiter applied to ARP packets that go to the routing engine. This prevents an ARP storm attack, either through misconfiguration or malicious behavior, from consuming all resources on the Routing Engine:

```
jweidley@MX240> show policer
Policers:
Name                                    Packets
__default_arp_policer__                 0
```

If the default policer does not meet your requirements, you have the ability to create a custom policer and apply it to the required interfaces. See the section *Protecting the Routing Engine* in Chapter 4 for more on creating and applying policers.

Chapter 4

Configuration Hardening

Configuration hardening is a process of enabling and disabling certain features to strengthen the security posture of a device. The Junos default configuration provides a reasonable level of security for most cases. But since it is possible to improperly configure a secure device in such a way that makes it less secure and reliable, this chapter covers the proper configuration of the various security options available for hardening the different aspects of a Junos device.

Network Hardening

First let's focus on understanding basic network behavior for services of Junos devices and how to harden them to provide predictable, reliable, and secure operation.

AutoInstallation

Autoinstallation is the automatic configuration of a device over the network from a preexisting configuration file that you create and store on a configuration server. You can use autoinstallation to automatically configure new devices and to deploy multiple devices from a central location in the network.

Some of the security concerns with autoinstallation is that the configuration servers use protocols that are not encrypted (i.e. TFTP, FTP, HTTP), which makes your configuration suseptable to sniffing, spoofing, and hijacking attacks. Also the security of your network is relying on the security posture of that configuration server. If someone breaks into the configuration server they could have complete control over your device configurations.

If autoinstallation is not desired in your environment it should be disabled.

Disabling Autoinstallation Completely

```
[edit]
jweidley@ex3200# delete system autoinstallation
```

Disabling Autoinstallation from USB (J-Series and SRX)

```
[edit]
jweidley@srx210# set system autoinstallation usb disable
```

TIP If autoinstallation would be beneficial in your network, it is highly recommended that an out-of-band management network that is isolated from users' access is used to reduce the probability of the previously mentioned security concerns.

Address Selection

By default, when traffic (cflowd, syslog, NTP, etc.) is generated from the routing engine, the source address is the IP address of the interface that is closest to the destination.

In larger, more diversely connected networks, outages could introduce variation in the expected source address selection behavior. To increase the predictability of the source address of traffic generated by the routing engine, it is recommended that the loopback (lo0) address be used.

Use the `default-address-selection` CLI option to instruct the routing engine to use the loopback interface (lo0) as the source address for all locally generated IP packets when the packet is sent through a routed interface. It's important to note that this does not apply when traffic is sent via the built-in out-of-band management interface (fxp0 or me0):

```
[edit]
jweidley@ex3200# set system default-address-selection
```

Many services (NTP, SNMP, RADIUS, etc.) can be configured with a source-address option which allows you to statically configure the source address to use for communication. In those circumstances the source address becomes the one that is specified with the source-address argument (provided the address is a valid address specified on the interface of a router), otherwise default-address-selection influences the default source address selection.

TIP Another benefit of using the loopback as the source address is that if all of your network devices have a loopback IP address that's in the same network range then you can easily create network firewall rules to clearly identify and permit management traffic from your devices.

CAUTION It is advisable to add this command during the initial configuration of your Junos device. Careful consideration should be taken when adding this command to a production device, and you should ensure all routing engine-generated traffic continues to work as expected in your environment after doing so.

ICMP Redirects

An ICMP redirect is a notification legitimately sent by a router to the sender of an IP packet to inform them of a better way to reach a particular destination host or network. Upon receipt of the redirect, the source device should modify their routing table and then route all subsequent packets through the newly suggested router.

TIP ICMP redirects should be the exception and not the rule. A properly designed network should not rely on ICMP to function normally.

The threat posed by ICMP redirects is that a DoS attack could be launched that forces a router to respond to thousands of suboptimally routed packets per second, consuming all valuable resources.

The default behavior of the Junos OS is to send ICMP redirects; but to globally disable ICMP redirects use the following:

```
[edit]
jweidley@ex3200# set system no-redirects
```

There are situations when redirects may need to be enabled to keep the network running for a short period of time until a permanent fix can be designed and implemented. For these situations, engineers can disable redirects on some interfaces and leave redirects enabled on other interfaces. Use the command below to disable ICMP redirects on a per interface basis:

```
[edit interfaces ge-0/0/3 unit 50]
jweidley@ex3200# set family inet no-redirects
```

Ping Timestamp and Record Route

It is a recommended practice to block pings from external sources, but ping is a very helpful troubleshooting tool for internal hosts. A common practice is to rate-limit ICMP traffic to the routing engine so you can safely provide this functionality without subjecting the Routing Engine to a possible DoS. (See the section in this chapter on *Protecting the Routing Engine* for more details.)

When the ping command is used with the record-route option, the Routing Engine displays the path of the ICMP echo request packets and timestamps in the ICMP echo responses. This is useful for troubleshooting network path problems because, unlike the traceroute command, it also shows the return route instead of just the path to the destination.

The possible security implication is that the timestamp and record route option allows someone to map your network and reveal private information, such as loopback addresses. Use the commands below to disable the ping, record-route, and timestamp options:

```
[edit]
jweidley@ex3200# set system no-ping-record-route
[edit]
jweidley@ex3200# set system no-ping-time-stamp
```

MORE? For more information about ping, record-route, and timestamp functionalities, see *This Week: A Packet Walkthrough on M, MX, and T Series* by Antonio Sanchez-Monge at http://www.juniper.net/dayone.

ICMP Source Quench

ICMP source quench (ICMP Type 4) is a congestion control technique used by receiving devices to tell the sending the device to reduce the amount of traffic it is sending. Ideally it is used to control the flow of traffic to reduce retransmissions and dropped packets but in reality it can be used in a blind throughput reduction attack.

Routers shouldn't process source quench messages (RFC1812) and RFC6633 formally deprecates its handing in other transport protocols. There are more effective methods of implementing congestion control and ICMP source quench has been found to be largely ineffective and for those reasons source quench should be disabled:

```
[edit]
jweidley@ex3200# set system internet-options no-source-quench
```

ICMP Rate Limiting

ICMP is one of the main protocols of the Internet Protocol (IP) suite. It is used for troubleshooting and to communicate error conditions in the network. Since the inception of the Internet there have been many different attacks and DoS that leveraged ICMP in unexpected ways. For this reason, the ICMP protocol suite should be appropriately restricted and rate-limited to allow ICMP to do its job while limiting potential risk.

This section involves limiting the number of ICMP messages that the Routing Engine can generate and/or process per second. KB28184 tells us that the Junos default is 1000 ICMP messages per second. If this value seems aggressive, you can use the `icmpv4-rate-limit` option to optimize the behavior for your network:

```
[edit]
jweidley@ex3200# set system internet-options icmpv4-rate-limit packet-rate 100
```

TIP Use the `icmpv6-rate-limit` option to change the handling of ICMPv6 messages.

SYN-FIN TCP Flags

One of the ways to determine the operating system type and version is to send non-standard packet types and analyze the target's response. This technique is referred to as TCP/IP Stack Fingerprinting. One of the common packet types used in the process is a TCP packet with both the SYN and FIN flags set. This is obviously an invalid packet since the SYN flag is used during the initial set up of a connection and the FIN flag is used when a session is being closed.

Packets with the TCP SYN-FIN flags set can also be used for other nefarious purposes and should be dropped. Although this invalid TCP flag combination could be blocked using a firewall filter, Junos also has an option under `[system internet-options]` hierarchy to drop these packets at the kernel level:

```
{master:0}[edit]
jweidley@EX4500# set system internet-options tcp-drop-synfin-set
```

TCP Reset (RST) Packets

The common response to a request for service on a closed port is to send a RST or a RST/ACK. Portscanning utilities send packets to network devices on a range of ports and listen to their response to determine if there is a service running on that port.

Aggressive scanning activity can put an unnecessary load on the Routing Engine and could negatively impact performance or cause a DoS. Use the `no-tcp-reset` option to tell Junos not to send TCP reset (RST) packets for connections made to non-listening ports. Here, let's use the CLI to view possible completions:

```
[edit]
jweidley@mx240# edit system internet-options

[edit system internet-options]
jweidley@mx240# set no-tcp-reset ?
Possible completions:
  drop-all-tcp       Drop all TCP Packets
  drop-tcp-with-syn-only  Drop only those TCP Packets with SYN bit
```

You can see the options:

- `drop-all-tcp`: Detects and drops packets with non-standard TCP flag combinations such as FIN/RST, ACK/RST, etc.
- `drop-tcp-with-syn-only`: Detects and drops TCP packets with only the SYN flag set or flag combinations that include the SYN flag.

Link Layer Discover Protocol (LLDP)

Link Layer Discover Protocol (LLDP) is a vendor-neutral standard Layer 2 protocol that allows network devices to advertise their identity and capabilities on the LAN. LLDP-capable devices transmit information in the form of type, length, and value (TLVs). These messages can include device-specific information such as chassis and port information, system name, and device capabilities.

LLDP-Media Endpoint Discovery (LLDP-MED) is an extension of the LLDP standard that supports interoperability between Voice Over IP (VoIP) endpoints and other networking end devices. LLDP-MED uses additional TLV information such as network policy discovery and Power over Ethernet (PoE) management.

LLDP and LLDP-MED have different configuration stanzas in the Junos OS but throughout this section both are simply referred to as LLDP.

The security risk with LLDP is that packets are not authenticated or encrypted, meaning that a malicious user could use techniques to conduct reconnaissance and gain information about your network. It may also be possible for an attacker to inject bogus LLDP packets in an attempt to find vulnerabilities or crash the device and create a DoS attack.

On the EX Ethernet Series switching platforms, LLDP is enabled on all interfaces by default. The recommended way to secure LLDP is to only enable it on those specific interfaces that require the functionality (router facing interfaces, switch facing interfaces, VoIP phones, etc.). Figure 4.1 has a sample network topology with LLDP restricted to desired interfaces.

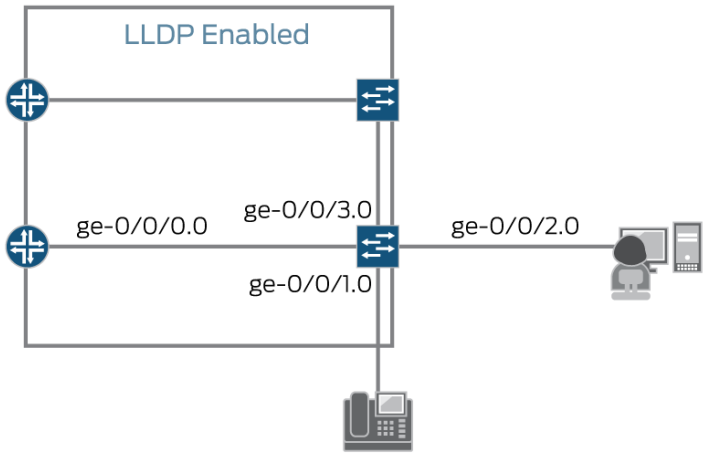


Figure 4.1 Recommended LLDP Deployment

Using the topology in Figure 4.1, let's first disable LLDP on all interfaces, and then specifically define the interfaces that should have LLDP enabled as more specific interface configurations to override the default `disable`:

```
[edit]
jweidley@ex3200# edit protocols lldp

[edit protocols lldp]
jweidley@ex3200# set interface all disable

[edit protocols lldp]
jweidley@ex3200# set interface ge-0/0/0.0

[edit protocols lldp]
jweidley@ex3200# set interface ge-0/0/3.0

[edit protocols lldp]
jweidley@ex3200# top edit protocols lldp-med

[edit protocols lldp-med]
jweidley@ex3200# set interface ge-0/0/1.0
```

Now verify the LLDP complete configuration:

```
[edit protocols]
jweidley@ex3200# show
lldp {
    interface all {
        disable;
    }
    interface ge-0/0/0.0;
    interface ge-0/0/3.0;
}
lldp-med {
    interface all {
        disable;
    }
    interface ge-0/0/1.0;
}
```

NOTE The LLDP and LLDP-MED features were introduced on the EX Ethernet Switching platforms but have also been implemented in the MX and branch SRX platforms. Check Juniper's online documentation to see if LLDP and LLDP-MED are supported on your specific platforms.

Try it Yourself: What Information Gets Shared via LLDP

Let's look at what kind of information can be obtained via LLDP by using the `show lldp neighbors interface <interface>` command. Try it with a lab device:

```
jweidley@srx210> show lldp neighbors interface fe-0/0/7.0
LLDP Neighbor Information:
Local Information:
Index: 1 Time to live: 120 Time mark: Tue Mar  1 22:59:37 2011 Age: 0 secs
Local Interface    : fe-0/0/7.0
Parent Interface   : -
Local Port ID      : 531
Ageout Count       : 0
Neighbor Information:
Chassis type       : Mac address
Chassis ID         : 00:23:9c:01:96:80
Port type          : Locally assigned
Port ID           : 531
Port description   : Connection to SRX210 DMZ Port
System name        : ex3200

System Description : Juniper Networks, Inc. ex3200-24t , version 10.4R3.4 Build date: 2010-08-13
12:56:38 UTC
System capabilities
       Supported  : Bridge Router
       Enabled    : Bridge Router
Management Info
       Type            : IPv4
       Address          : 192.168.5.1
Port ID         : 0
       Subtype          : 1
       Interface Subtype : Unknown(1)
       OID             : 1.3.6.1.2.1.31.1.1.1.1.0
```

From the bolded segments of the output you can tell that this SRX210 is connected to an EX3200, that it has 24 copper ports, that it is running Junos 10.4R3.4, and that it has a management address of 192.168.5.1.

Management Services

Network devices can provide a wealth of information about network utilization, errors, and traffic patterns. The information obtained from network devices can also be used for capacity planning, incident response, fault isolation, and forensics. This information needs to be accurate, timely, and obtained in a secure manner.

This section covers secure methods of gathering information, backing up device configuration files, setting alerts based on storage utilization, and how to maintain secure and accurate time.

Network Time Protocol (NTP)

From a security auditing perspective, accurate time is imperative so engineers can correlate system events to gather the root cause of problems. Network Time Protocol (NTP) is an industry standard for synchronizing time between devices to a common reference clock. NTP communicates via UDP and could be a security risk if a malicious user was to spoof the IP address of the NTP server and inject an inaccurate timestamp. To mitigate this potential risk, NTP should be secured by implementing authentication.

By default, NTP is not enabled in Junos. The steps below show how to enable NTP with authentication using MD5 as the hashing mechanism.

MORE? For more detailed NTP information see the *Junos OS Basic System Configuration Guide* at http://www.juniper.net/documentation/en_US/junos14.2/topics/task/configuration/network-time-protocol-time-server-time-services-configuring.html.

How to Enable NTP with MD5 Authentication

1. Start with a basic NTP configuration:

```
[edit]
jweidley@ex3200# set system time-zone America/New_York
[edit]
jweidley@ex3200# edit system ntp
[edit system ntp]
jweidley@ex3200# set boot-server 192.168.3.2
```

2. Configure the authentication key. Authentication key ID and value must match among NTP clients and servers. Choose a shared secret password that follows a password complexity policy:

```
[edit system ntp]
jweidley@ex3200# set authentication-key 1 type md5 value Z3l>L8@w
```

3. Set the configured authentication key ID as "trusted":

```
[edit system ntp]
jweidley@ex3200# set trusted-key 1
```

4. Since accurate time is critical, configure primary and secondary NTP servers and reference the trusted key ID. To increase resiliency, ensure the secondary NTP server is located on a different network than the primary server:

```
[edit system ntp]
jweidley@ex3200# set server 192.168.3.2 key 1 prefer
[edit system ntp]
jweidley@ex3200# set server 192.168.33.2 key 1
```

5. (Optional) If `set system default-address-selection` is not configured, use this command to force all NTP communications to use the loopback address:

```
[edit system ntp]
jweidley@ex3200# set source-address 172.24.3.1
```

6. Verify the configuration:

```
[edit system ntp]
jweidley@ex3200# show
boot-server 192.168.3.2;
authentication-key 1 type md5 value " $9$-kboZjHqKvMWLNs4"; ## SECRET-DATA
server 192.168.3.2 key 1 prefer; ## SECRET-DATA
server 192.168.33.2 key 1; ## SECRET-DATA
trusted-key 1;
source-address 172.24.3.1;
```

7. Commit and comment the configuration:

```
[edit system ntp]
jweidley@ex3200# commit and-quit comment "Configured NTP Auth"
commit complete
Exiting configuration mode
jweidley@ex3200>
```

8. Force an update and verify proper operation:

```
jweidley@ex3200> set date ntp
10 Mar 17:30:34 ntpdate[27784]: step time server 192.168.3.2 offset 0.000603 sec
```

9. Now that NTP is set up and operational, the next step to further secure your NTP configuration is to set a firewall filter to limit NTP traffic to trusted sources. See the sample filter firewall below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit NTP:

```
set firewall family inet filter protect-re term allow-ntp from source-address <SERVER>
set firewall family inet filter protect-re term allow-ntp from source-address 127.0.0.0/24
set firewall family inet filter protect-re term allow-ntp from protocol udp
set firewall family inet filter protect-re term allow-ntp from destination-port ntp
set firewall family inet filter protect-re term allow-ntp then accept
```

CAUTION Cryptographic operations can be resource intensive, so even though a Junos device can be configured as an NTP source for other network devices, it's considered a best security practice to have a dedicated NTP server.

NOTE It's a best security practice to use an NTP server located on an out-of-band management network to keep management traffic and user traffic separated.

Simple Network Management Protocol Version 2 (SNMPv2)

SNMP is a widely accepted standard protocol for remotely monitoring and managing of your network devices. SNMP provides the capability to gather topological and utilization data about the network for troubleshooting and planning purposes. Although SNMP is a very useful tool, earlier versions were inherently insecure because they used a simple community-based access control mechanism. Access is

controlled by using a simple password, called a *community string*. The community string is sent between devices unencrypted and is susceptible to snooping attacks. SNMP's security reputation has also been made worse by vendors that enable the protocol by default with standard or easily guessable community strings.

Security can be greatly increased by using an out-of-band management network where all SNMP communications are kept separate from user traffic.

In the Junos OS, SNMP is not enabled by default and does not have a default community string. The Junos SNMP implementation also includes many features to separate and limit access that are configured in this section.

TIP It's a best security practice to use complex community strings and to change them periodically since they are transmitted in plain-text and susceptible to capture.

MORE? For a detailed explanation of SNMP, and to review the SNMP samples, see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

NOTE There are references to specific MIB names used in the configuration examples throughout this *SNMP* section. MIBs can be specific to the version of Junos, but a complete list of MIB names for Junos 12.3 can be found at http://www.juniper.net/techpubs/en_US/junos12.3/topics/concept/juniper-specific-mibs-junos-nm.html or http://www.juniper.net/techpubs/en_US/junos12.3/information-products/pathway-pages/mibs-and-traps-ref/mibs-and-traps-ref.pdf.

How to Enable SNMPv2

1. Set the system location to something descriptive enough for someone to find the device:

```
[edit snmp]
jweidley@ex3200# set location "DC1-Rack:8-Row:2"
```

2. Set the system contact to something descriptive so the NOC knows whom to contact in case of an emergency or problems with this device:

```
[edit snmp]
jweidley@ex3200# set contact "CompanyName NOC:123.456.7890"
```

3. Define the community string. Since the community string is essentially a password, it is common practice for the community string to follow the password complexity policy:

```
[edit snmp]
jweidley@ex3200# edit community S8M!y:4b
```

4. Only permit `read-only` mode to eliminate any possibility of changes to the few writable MIBs:

```
[edit snmp community "S8M!y:4b"]
jweidley@ex3200# set authorization read-only
```

5. Define primary and secondary SNMPv2 servers that are allowed to send queries. To increase resiliency, ensure the secondary SNMP server is located on a different network than the primary server. Then add the recommended `default restrict` option to deny access to all SNMP clients that are not explicitly listed:

```
[edit snmp community "S8M!y:4b"]
jweidley@ex3200# set clients 192.168.3.3/32

[edit snmp community "S8M!y:4b"]
jweidley@ex3200# set clients 192.168.33.3/32

[edit snmp community “S8M!y:4b”]
jweidley@ex3200# set clients default restrict
```

6. Verify the configuration:

```
[edit snmp]
jweidley@ex3200# show
location DC1-Rack:8-Row:2;
contact CompanyName NOC:123.456.7890;
community S8M!y:4b {
   authorization read-only;
   clients {
      192.168.3.3/32;
      192.168.33.3/32;
      0.0.0.0/0 restrict;
   }
 }
```

7. Commit and comment the configuration:

```
[edit snmp]
jweidley@ex3200# commit and-quit comment "Configured SNMPv2"
commit complete
Exiting configuration mode

jweidley@ex3200>
[edit snmp]
```

8. (Optional) If SNMP queries are not received via an out-of-band (OOB) network, overall security can be increased by restricting SNMP queries to specific interfaces:

```
jweidley@ex3200# set interface ge-0/0/1.0
```

9. Now that SNMP is set up, the next step to further secure your SNMP configuration is to set a firewall filter to limit SNMP traffic to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit SNMP:

```
set firewall family inet filter protect-re term allow-snmp from source-address <CLIENT>
set firewall family inet filter protect-re term allow-snmp from protocol udp
set firewall family inet filter protect-re term allow-snmp from destination-port snmp
set firewall family inet filter protect-re term allow-snmp then accept
```

How to Use SNMP Views

In the preceding configuration, the defined management stations are able to access the entire MIB on the device. One drawback is that there could be a performance impact if an authorized management station queries every MIB object repeatedly.

One way to prevent this type of unintentional or inadvertent behavior is to limit management stations to only the MIB objects that they need. This is done by creating a view. Creating a view takes a bit of research to find out exactly which MIBs the management station needs access to.

In this example let's assume the management stations only need to retrieve chassis inventory information. Here's how to add a view to the preceding SNMPv2 configuration:

1. Create a view that includes only the desired MIBs:

```
[edit snmp]
jweidley@ex3200# set view inventory-only oid jnxBoxAnatomy include

[edit snmp]
jweidley@ex3200# set view inventory-only oid system include
```

2. Associate the view with the desired community string:

```
[edit snmp]
jweidley@ex3200# set community S8M!y:4b view inventory-only
```

3. Verify configuration:

```
[edit snmp]
jweidley@ex3200# show
location DC1-Rack:8-Row:2;
contact CompanyName NOC:123.456.7890;
view inventory-only {
   oid jnxBoxAnatomy include;
   oid system include;
}
community S8M!y:4b {
   view inventory-only;
   authorization read-only;
   clients {
      192.168.3.3/32;
      192.168.33.3/32;
      0.0.0.0/0 restrict;
   }
 }
```

Using Multiple SNMP Communities

With a single community, all management stations get access to the same information using the same community string. In some scenarios, this may not be desirable. But using SNMP views and multiple communities allows you to compartmentalize access to specific MIBs with different community strings (passwords).

In the next configuration example let's use the `client-list` CLI option in order to group SNMP management stations from different organizations. The *performance* client-list is for the service provider's SNMP stations and they get access to anatomy, interface, OSPF, and BGP MIBs, while the *partner* client-list defines your partner's SNMP stations and is restricted to only the interface MIBs.

1. Create the view called `system-level` for the service provider's performance servers:

```
[edit snmp]
jweidley@MX80# set view system-level oid jnxBoxAnatomy include

[edit snmp]
jweidley@MX80# set view system-level oid 1.3.6.1.2.1.2 include

[edit snmp]
jweidley@MX80# set view system-level oid 1.3.6.1.2.1.14 include
```

```
[edit snmp]
jweidley@MX80# set view system-level oid 1.3.6.1.2.1.15 include
```

2. Create the view called `limited` for the partner's management servers:

```
[edit snmp]
jweidley@MX80# set view limited oid 1.3.6.1.2.1.2 include
```

3. Create the client-list for the service provider's performance servers:

```
[edit snmp]
jweidley@MX80# set client-list performance 192.168.10.0/28

[edit snmp]
jweidley@MX80# set client-list performance 192.168.20.0/28

[edit snmp]
jweidley@MX80# set client-list performance default restrict
```

4. Create the client-list for the partner's servers:

```
[edit snmp]
jweidley@MX80# set client-list partner 172.16.1.0/28

[edit snmp]
jweidley@MX80# set client-list partner 172.16.10.0/28

[edit snmp]
jweidley@MX80# set client-list partner default restrict
```

5. Create the first community string, enable read-only access, and associate the appropriate client-list and view for the service provider:

```
[edit snmp]
jweidley@MX80# set community CfL!d4#2 authorization read-only

[edit snmp]
jweidley@MX80# set community CfL!d4#2 client-list-name performance

[edit snmp]
jweidley@MX80# set community CfL!d4#2 view system-level
```

6. Create the second community string, enable read-only access, and associate the appropriate client-list and view for the partner:

```
[edit snmp]
jweidley@MX80# set community xH#5^Gp9 authorization read-only

[edit snmp]
jweidley@MX80# set community xH#5^Gp9 client-list-name partner

[edit snmp]
jweidley@MX80# set community xH#5^Gp9 view limited
```

7. Verify the configuration:

```
[edit snmp]
jweidley@MX80# show
location DC1-Rack:5-Row:2;
contact "CompanyName NOC:123.456.7890";
view system-level {
   oid jnxBoxAnatomy include;
```

```
    oid 1.3.6.1.2.1.2 include;
    oid 1.3.6.1.2.1.14 include;
    oid 1.3.6.1.2.1.15 include;
}
view limited {
    oid 1.3.6.1.2.1.2 include;
}
client-list performance {
    192.168.10.0/28;
    192.168.20.0/28;
}
client-list partner {
    172.16.1.0/28;
    172.16.10.0/28;
    0.0.0.0/0 {
          restrict;
    }

}
community "CfL!d4#2" {
    view system-level;
    authorization read-only;
    client-list-name performance;
}
community "xH#5^Gp9" {
    view limited;
    authorization read-only;
    client-list-name partner;
}
```

8. Commit and comment the configuration change:

```
[edit]
jweidley@MX80# commit and-quit comment "enabled multiple snmp communities"
commit complete
Exiting configuration mode

jweidley@MX80>
```

Simple Network Management Protocol Version 3 (SNMPv3)

As previously discussed, the major drawback with SNMP v1 and v2 is that all communications are unencrypted and unauthenticated. This means that anyone who can reach the device via the network could send queries, attempt to guess the community string, or they could intercept information to gain a detailed understanding of your network.

SNMPv3 increases security by using a user-based security model (USM) to authenticate and encrypt SNMP communications. The Junos SNMPv3 implementation supports various hashing and encryption algorithms – obviously the strongest methods provide maximum security but you have to choose algorithms and encryption that are supported by your management stations.

Before starting with the sample configurations, it's important to discuss the SNMPv3 engine ID. The engine ID should be a unique value that is used to identify the device to the SNMP manager. By default, the local engine ID uses the default IP address of the device. There may be situations where you want to hard code a different value to accommodate your SNMP management stations or a unique deployment scenario.

CAUTION SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID. If you change the engine ID, you'll have to change the SNMPv3 user passwords, as they are based on the previous engine ID and you'll receive a warning while committing your configuration. It's best to configure and commit the engine ID first and then configure your SNMPv3 users.

MORE? For additional information regarding the engine ID see Juniper Networks Technical Documentation at http://www.juniper.net/techpubs/en_US/junos14.1/topics/task/configuration/local-engine-id-configuring-junos-nm.html.

MORE? Also for a detailed explanation of SNMP and to review sample SNMP configurations, see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

How to Set Up a USM for SNMPv3 Communications

1. Set the engine ID to the MAC address of the management interface and then commit the configuration:

```
[edit snmp]
jweidley@srx210# set engine-id use-mac-address

[edit snmp]
jweidley@srx210# commit
commit complete

[edit]
jweidley@srx210#
```

2. Create a restricted *view* to limit what MIBs that this SNMPv3 user can access:

```
[edit snmp]
jweidley@srx210# set view inventory-view oid jnxBoxAnatomy include

[edit snmp]
jweidley@srx210# set view inventory-view oid system include
```

3. Create SNMPv3 user accounts, select authentication and privacy algorithms, and choose passwords. In this example, the most secure hashing and encryption methods are used to provide maximum security. It's important to note that the authentication and privacy passwords should be different and unique and should follow your organization's password complexity policy:

```
[edit snmp]
jweidley@srx210# edit v3

[edit snmp v3]
jweidley@srx210# set usm local-engine user nms-user authentication-sha authentication-password
S8M!y:4b

[edit snmp v3]
jweidley@srx210# set usm local-engine user nms-user privacy-aes128 privacy-password $Y5wIm@4
```

4. VACM stands for View-based Access Control Model and this is where you tie together the configuration statements from Steps 1-3 to grant access:

```
[edit snmp v3]
jweidley@srx210# edit vacm
```

```
[edit snmp v3 vacm]
jweidley@srx210# set security-to-group security-model usm security-name nms-user group inventory-
view

[edit snmp v3 vacm]
jweidley@srx210# set access group inventory default-context-prefix security-model usm security-
level privacy read-view inventory-view

[edit snmp v3 vacm]
jweidley@srx210# set access group inventory default-context-prefix security-model usm security-
level privacy notify-view inventory-view
```

5. Review the configuration:

```
[edit snmp v3 vacm]
jweidley@srx210# up 2

[edit snmp]
jweidley@srx210# show
v3 {
    usm {
        local-engine {
            user nms-user {
                authentication-sha {
                    authentication-key "$9$BcnEreMWxws48Lk.P5F39ApOBEcylvMXn/u1IEyrvWLxbsUjH.
mTJZ9AtpB1X7Nb4aiHmf5FmP39CAOONdVsYojHqzn/goDk.mTQcyrlWLs24aJD4oUHmfzFn/CABIylK8xN69OIESeKoJZUDkm
PQF39HkIEhSMWaZGUqm69ABRhOO1hSyW8-VwY2a"; ## SECRET-DATA
                }
                privacy-aes128 {
                    privacy-key "$9$yg6eK87NbY4aSrWxN-wsz3nC0BcylKvLREgoaGiHfTz3nCOORSyKuOhrKMN-.
Pfz9AOBEeK836KMLxdV24aZGik.PTQnVwz369OBSreMWxws4GUHaJ36AtIRlKMW7-s24aGDre24oJHkIEhrM8-Vwsgo8Lik.
mF3hSyKvLxNdb24LXHq.fzF69AuIE"; ## SECRET-DATA
                }
            }
        }
    }
    vacm {
        security-to-group {
            security-model usm {
                security-name nms-user {
                    group inventory-view;
                }
            }
        }
        access {
            group inventory {
                default-context-prefix {
                    security-model usm {
                        security-level privacy {
                            read-view inventory-view;
                            notify-view inventory-view;
                        }
                    }
                }
            }
        }
    }
}
engine-id {
    use-mac-address;
}
view inventory-view {
    oid jnxBoxAnatomy include;
    oid system include;
}
```

6. Now that SNMPv3 is set up, the next step to further secure your SNMP configuration is to set a firewall filter to limit SNMP traffic to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit SNMP:

```
set firewall family inet filter protect-re term allow-snmp from source-address <CLIENT>
set firewall family inet filter protect-re term allow-snmp from protocol udp
set firewall family inet filter protect-re term allow-snmp from destination-port snmp
set firewall family inet filter protect-re term allow-snmp then accept
```

SNMPv3 Traps

SNMP traps are unsolicited messages sent by network devices to notify management stations of important events. SNMP traps were not discussed in the previous section because other than a strong community string and using an out-of-band management network, there's not much else in the way of security features to configure.

The cryptographic security and USM described above also apply to generating SNMPv3 traps, but in this example let's make the necessary configuration statements to allow your Junos device to send SNMPv3 traps to an SNMPv3 management station.

Since the SNMPv3 configuration is somewhat difficult to read, let's remove the previous SNMPv3 configuration and configure SNMPv3 traps from scratch.

CAUTION SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID. If you change the engine ID, you'll have to change the SNMPv3 user passwords, as they are based on the previous engine ID and you'll receive a warning while committing your configuration. It's best to configure the engine ID first and then configure your SNMPv3 users.

1. Set the engine ID to the MAC address of the management interface:

```
[edit snmp]
jweidley@srx210# set engine-id use-mac-address

[edit snmp] jweidley@srx210# commit commit complete

[edit]
jweidley@srx210#
```

2. Create SNMPv3 user accounts, select authentication and privacy algorithms, and choose passwords. In this example, let's use the most secure hashing and encryption methods to provide maximum security. It's important to note that the authentication and privacy passwords should be different and unique and should follow your organization's password complexity policy:

```
[edit snmp]
jweidley@srx210# edit v3

[edit snmp v3]
jweidley@srx210# set usm local-engine user nms-user authentication-sha authentication-password
S8M!y:4b

[edit snmp v3]
jweidley@srx210# set usm local-engine user nms-user privacy-aes128 privacy-password $Y5wIm@4
```

3. SNMP supports multiple types of notifications. In this example our notification method is `trap` and we'll create a reference tag, called *chassis-trap-receivers* that will be used in Step 5:

```
[edit snmp v3]
jweidley@srx210# set notify chassis-trap-list type trap

[edit snmp v3]
jweidley@srx210# set notify chassis-trap-list tag chassis-trap-receivers
```

4. Define the specific Trap MIBs to send to managers:

```
[edit snmp v3]
jweidley@srx210# set notify-filter chassis-traps oid jnxChassisTraps include

[edit snmp v3]
jweidley@srx210# set notify-filter chassis-traps oid jnxChassisOKTraps include
```

5. Specify the IP address of the SNMP manager (target) that will receive traps. Then link the server to the tag list in Step 3 and the SNMPv3 specific parameters in Step 6:

```
[edit snmp v3]
jweidley@srx210# edit target-address nms1

[edit snmp v3 target-address nms1]
jweidley@srx210# set address 192.168.3.2

[edit snmp v3 target-address nms1]
jweidley@srx210# set tag-list chassis-trap-receivers

[edit snmp v3 target-address nms1]
jweidley@srx210# set target-parameters noc-snmpv3-settings
```

6. Configure the SNMPv3 security parameters when talking to the network management servers:

```
[edit snmp v3 target-address nms1]
jweidley@srx210# up

[edit snmp v3]
jweidley@srx210# edit target-parameters noc-snmpv3-settings

[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# set parameters message-processing-model v3

[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# set parameters security-model usm

[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# set parameters security-level privacy

[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# set parameters security-name nms-user

[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# set notify-filter chassis-traps
```

7. Review the configuration:

```
[edit snmp v3 target-parameters noc-snmpv3-settings]
jweidley@srx210# up 2
```

```
[edit snmp]
jweidley@srx210# show
v3 {
   usm {
      local-engine {
         user nms-user {
            authentication-sha {
               authentication-key "$9$jQq5Q3nCB1h6/8X7-ws4aZUjqmfTF39YgGiHqf5Fn/
CO1evWXxdyr4aJZji9AtOhSMWxN-wx7s4oaUDtuO1IcvWLbYgEcK8XxdVmf5Tn/1RhSyKhceWxNbwYgoajHfTz6Ct24UHqPQ
zcyreK8x7Vws4W8Hq.P3nSrleLx24ajk.UDi.Pfn6pOBIRS"; ## SECRET-DATA
            }
            privacy-aes128 {
               privacy-key "$9$gnaJDkqfQ3624UHq.5TyleW7-YgoJZjbwFn6CuOEcyleW7Nb2gJx7s4JGq.1REyM
8N-waJDlKJGjHmPz369CuO1RcSeP5ylKMN-24aGUH5T3CAO6/lK8LVboJGUk.Tz36Ct4az3n/OOVws4GD.
P5TFnDjuO1Irls2gJZjHqmfz3jiOB1EyrKM8xVw"; ## SECRET-DATA
            }
         }
      }
   }
   target-address nms1 {
      address 192.168.3.2;
      tag-list chassis-trap-receivers;
      target-parameters noc-snmpv3-settings;
   }
   target-parameters noc-snmpv3-settings {
      parameters {
         message-processing-model v3;
         security-model usm;
         security-level privacy;
         security-name nms-user;
      }
      notify-filter chassis-traps;
   }
   notify chassis-trap-list {
      type trap;
      tag chassis-trap-receivers;
   }
   notify-filter chassis-traps {
      oid jnxChassisTraps include;
      oid jnxChassisOKTraps include;
   }
}
engine-id {
   use-mac-address;
}
```

For brevity, only a single trap receiver was configured, but for resiliency purposes multiple receivers should be configured under the `target-address` stanza in Junos.

Syslog

Syslog is an industry standard method for logging system information either locally, or to designated, remote servers. Logging is critical to device security because it creates an audit trail of system activity that can assist you in identifying configuration errors, investigating intrusions, troubleshooting service disruptions, and reacting to probes and scans.

NOTE Junos device platforms have a reasonable amount of drive space that can be used for local log storage. This enhances security by being able to retain logs locally on the device in case the Syslog server is unavailable or drops logs.

A general rule of thumb is that remote logs are used for forensic purposes and local logs are used for troubleshooting. You can also create separate local log files that contain different types of log messages. If desired you can also permit or deny access to specific logs from different user groups (auditors, operations, etc.) using login class permissions.

In the following configuration let's configure both local and remote Syslog, as well as separate specific message types in different log files.

1. Logged in users should receive any emergency message while they are logged in to the device:

```
[edit]
jweidley@EX3200# edit system syslog

[edit system syslog]
jweidley@EX3200# set user * any emergency
```

2. Any informational messages should be logged to the *messages* file locally on the device:

```
[edit system syslog]
jweidley@EX3200# set file messages any info

[edit system syslog]
jweidley@EX3200# set file messages authorization info
```

3. Let's create a separate file, called *User-Auth*, which contains all authorization information as well as any command issued by a logged-in user:

```
[edit system syslog]
jweidley@EX3200# set file User-Auth authorization any

[edit system syslog]
jweidley@EX3200# set file User-Auth interactive-commands any
```

4. Now let's create another local file, called *audit*, which contains all commands issued by a logged-in user:

```
[edit system syslog]
jweidley@EX3200# set file audit interactive-commands any
```

5. Now create another local file, called *processes*, which contains log messages generated by system daemons:

```
[edit system syslog]
jweidley@EX3200# set file processes daemon any
```

6. When connected on the console it's helpful to see system messages to be aware of the current status of the device. If you're connected to the console, you'll want this visibility:

```
[edit system syslog]
jweidley@EX3200# set console any any
```

7. It's recommended to send all messages to a remote Syslog server for auditing and forensic purposes. Be sure that you configure two Syslog servers for resiliency purposes:

```
[edit system syslog]
jweidley@EX3200# set host 192.168.3.2 any any

[edit system syslog]
jweidley@EX3200# set host 192.168.4.2 any any
```

8. By default the hostname is not included in Syslog messages that are sent to remote servers. To avoid confusion, it's a recommended practice to configure the `log-prefix` option with a unique identifier, like the hostname, in every Syslog message:

```
[edit system syslog]
jweidley@EX3200# set host 192.168.3.2 log-prefix EX3200

[edit system syslog]
jweidley@EX3200# set host 192.168.4.2 log-prefix EX3200
```

9. In some cases, the standard time format may not be as precise as you need it to be for computer forensic investigations or troubleshooting. Configure the `millisecond` and `year` options to make the timestamps be as precise as possible:

```
[edit system syslog]
jweidley@EX3200# set time-format millisecond year
```

10. (Optional) For consistency and resiliency, Syslog traffic should be sourced from the loopback address. Use this command if the `set system default-address` selection is not configured:

```
[edit system syslog]
jweidley@EX3200# set source-address 192.168.5.1
```

11. Review the configuration:

```
[edit system syslog]
jweidley@EX3200# show
user * {
    any emergency;
}
host 192.168.3.2 {
    any any;
    log-prefix EX3200;
}
host 192.168.4.2 {
    any any;
    log-prefix EX3200;
}
file messages {
    any info;
    authorization info;
}
file User-Auth {
    authorization any;
    interactive-commands any;
}
file audit {
    interactive-commands any;
}
file processes {
    daemon any;
}
console {
    any any;
}
time-format year millisecond;
```

MORE? See Knowledge Base Article 12679 for additional information regarding adding a hostname to remote Syslog messages: http://kb.juniper.net/InfoCenter/index?page=content&id=KB12679.

MORE? Another important topic that is beyond the scope of this book is log file management. The internal storage media can only hold so much data and you have to balance the amount of logs you need on the device and how much drive space you have. For examples of configurations for limiting the size and number of log files see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

MORE? For more information on planning and implementing Syslog, see the book, *Junos High Availability* (by James Sonderegger, Orin Blomberg, Kieran Milne & Senad Palislamovic, O'Reilly Media, 2009) at http://www.juniper.net/books.

Configuration Backups

Reliable configuration backups are essential for disaster recovery purposes and are perhaps one of the most important, yet most neglected, aspects of device administration.

The Junos OS keeps copies of previous configurations on the device by default, although the number varies from platform to platform. This feature makes it easy to recover quickly from misconfigurations, using the `rollback` feature, or to see what changed between commits, using the `compare` feature. But it's also recommended that device configurations be archived on an external system so they can be retrieved in the event of catastrophic device failure or device recovery. The archive server should be a hardened server with whole disk encryption and with access limited to only essential personnel.

There are a few open source and commercially available software packages to backup device configurations, but in this section we will focus on configuring two native Junos features to automatically archive your device configurations to a remote server.

MORE? For more information regarding the backup information covered in this section, visit: http://www.juniper.net/techpubs/en_US/junos14.1/topics/task/configuration/junos-software-system-management-router-configuration-archiving.html.

Configuring Periodic Configuration Backups

BBacking up configurations at consistent intervals is beneficial for devices whose configurations don't change that frequently.

So in this section let's configure the `transfer-interval` Junos feature where the configuration is archived every 24 hours (1440 minutes). Junos supports a few transport protocols but let's configure the Secure Copy Protocol (SCP) to ensure the confidentiality and integrity of the transfer.

1. Configure the `transfer-interval` option. Define the interval for which you want your configurations backed up in minutes (the available options are between 15 – 2880):

```
[edit]
jweidley@EX3200# edit system archival configuration
```

```
[edit system archival configuration]
jweidley@EX3200# set transfer-interval 1440
```

2. Enter the server username, hostname, directory, and password for the archive server. The format is very specific, so ensure it is entered correctly:

```
[edit system archival configuration]
jweidley@EX3200# set archive-sites scp://jweidley@192.168.3.2:/Configs password 3zP%a9@E
```

3. Junos makes an SSH connection to the archive server after you press the enter key, in order to capture the archive host's public key. You have to confirm, so proceed by typing *yes*:

```
The authenticity of host 192.168.3.2 (192.168.3.2) cant be established.
RSA key fingerprint is 84:da:22:78:d2:26:df:86:e5:1f:c0:33:41:db:35:02.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 192.168.3.2 (RSA) to the list of known hosts.

[edit system archival configuration]
jweidley@EX3200#
```

4. Review the configuration:

```
[edit system archival configuration]
jweidley@EX3200# show
transfer-interval 1440;
archive-sites {
    "scp://jweidley@192.168.3.2:/Configs" password "$9$EGCyMCVb1JGnev2aajPf359AO1"; ## SECRET-DATA
}
```

The public key of the archive server is automatically stored in the configuration under the `[security ssh-known-hosts]` Junos hierarchy for future use.

When the file is transferred to the archive server it will have a unique name that is made up of the device hostname, date, and time of the transfer:

```
<device-name>_juniper.conf[.gz]_YYYYMMDD_HHMMSS
```

CAUTION Step 2 requires the plain-text archive server password be entered as part of the command. Be sure you take the necessary steps to preserve the security of the password from shoulder surfers and console logging. To avoid hard coding the user's server password in your configuration use SSH keys. See the Junos documentation for more details, https://www.juniper.net/techpubs/en_US/junos14.1/topics/task/configuration/security-ssh-hostkeys-secure-copying.html.

TIP Be sure to review the logs (`show log messages`) after configuration archival is configured. Junos provides descriptive messages that indicate successful and failed transfers. Also use the `show system configuration archival` command to see files that are pending transfer.

Configuring On-Demand Configuration Backups

Periodic configuration backups may not be optimal for devices with frequent changes because you could lose configuration data if they're only being backed up every 24 hours.

Let's configure the `transfer-on-commit` Junos CLI feature where, after a commit is completed, the Junos OS transfers the configuration to an archive server. Now

backups are automatic whenever a configuration changes, not just on a time schedule. Junos supports using a few different transport protocols but let's configure SCP to ensure the confidentiality and integrity of the transfer.

1. First, enable the `transfer-on-commit` option:

```
[edit]
jweidley@EX3200# edit system archival configuration

[edit system archival configuration]
jweidley@EX3200# set transfer-on-commit
```

2. Enter the server username, hostname, directory, and password for the archive server. The format is specific so ensure it is entered correctly:

```
[edit system archival configuration]
jweidley@EX3200# set archive-sites scp://jweidley@192.168.3.2:/Configs password 3zP%a9@E
```

3. Junos makes an SSH connection to the archive server after you press the enter key, in order to capture the archive host's public key. You have to confirm proceeding by typing *yes*:

```
The authenticity of host 192.168.3.2 (192.168.3.2) cant be established.
RSA key fingerprint is 84:da:22:78:d2:26:df:86:e5:1f:c0:33:41:db:35:02.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 192.168.3.2 (RSA) to the list of known hosts.

[edit system archival configuration]
jweidley@EX3200#
```

4. Review the configuration:

```
[edit system archival configuration]
jweidley@EX3200# show
transfer-on-commit;
archive-sites {
    "scp://jweidley@192.168.3.2:/Configs" password "$9$EGCyMCVb1JGnev2aajPf359AO1"; ## SECRET-DATA
}
```

The public key of the archive server is automatically stored in the configuration under the `[security ssh-known-hosts]` Junos hierarchy for future use.

When the file is transferred to the archive server it will have a unique name that is made up of the device hostname, date, and time of the transfer:

```
<device-name>_juniper.conf[.gz]_YYYYMMDD_HHMMSS
```

CAUTION Step 2 requires the actual password be entered as part of the command. Ensure that you take the necessary steps to preserve the security of the password from shoulder surfers and console logging. To avoid hard coding the user's server password in your configuration is use SSH keys. See the Junos documentation for more details: https://www.juniper.net/techpubs/en_US/junos14.1/topics/task/configuration/security-ssh-hostkeys-secure-copying.html.

TIP Remember to review the logs (`show log messages`) after configuration archival is configured. Junos provides descriptive messages that indicate successful and failed transfers. Also use the `show system configuration archival` command to see files that are pending transfer.

Mass Storage

The Routing Engine on Junos devices is essentially a server with a Kernel, file system, processes, RAM and some kind of mass storage device (i.e. hard drive). That mass storage device can be used as a secondary boot device and to store log files, Junos images, automation scripts, etc. Storage space is a finite resource and it is possible for it to be consumed and impact operations of the device.

In this section we'll configure Junos to monitor the utilization of /var partition. When the `high` and `full` levels are reached a system alarm will be generated:

```
[edit]
jweidley@ex3200# set chassis disk-partition /var level high free-space 25 percent

[edit]
jweidley@ex3200# set chassis disk-partition /var level full free-space 10 percent

[edit]
jweidley@ex3200# show chassis
disk-partition /var {
    level high {
        free-space 25 percent;
    }
    level full {
        free-space 10 percent;
    }
}
```

TIP Use the `show system storage detail` command to see the partition sizes and how much space is available. When space is required, use the `request system storage cleanup` command to remove unnecessary files.

Access Security

There are a variety of ways to manage a Junos device. This section shows you how to disable insecure access services, how to enable secure access services, and how to set a warning banner.

By default, all access services are disabled and network engineers must enable which services are required. The exception to this is on the branch SRX platforms and some J Series routers, where some services are enabled by default on the "trust" zone.

NOTE The Junos OS supports other secure access services that are not covered in this section, but these services can also be hardened using the same concepts as the services defined below.

Warning Banner

One of the simplest things you can do is to configure a warning banner, the electronic equivalent of a "No Trespassing" sign. Although it doesn't technically protect your device or network, having a warning banner with very specific acceptable use instructions could help your organization by serving as a reminder of acceptable use and as a legal disclaimer.

To be most effective, the warning banner should be presented to the user prior to the user entering their login credentials.

Junos permits a warning banner of 2048 characters, which should be enough space to convey a detailed message of your expectations for the device.

It's easy enough to do. For example:

```
[edit]
jweidley@MX80# set system login message "\n\tUNAUTHORIZED USE OF THIS SYSTEM\n\tIS STRICTLY
PROHIBITED!\n\tPlease contact company-noc@company.com to gain access to this equipment if you need
authorization.\n"
```

And this login message configuration example would produce a login message similar to the following:

```
server% ssh router1
Trying 1.1.1.1...
Connected to router1.
Escape character is ^].

   UNAUTHORIZED USE OF THIS SYSTEM
   IS STRICTLY PROHIBITED!

   Please contact /company-noc@company.com/ to gain
   access to this equipment if you need authorization.

MX80 (ttyp0)
login:
```

MORE? For more information regarding login message formatting, visit http://www.juniper.net/techpubs/en_US/junos14.1/topics/task/configuration/authentication-router-login-message.html.

TIP The login message is displayed for remote access connection, such as telnet and SSH, but it also is displayed for J-Web connections. Be aware that you might have do some formatting so that the banner is easily readable from all access services.

Disable Insecure Access Services

Access services are considered insecure when communication to the device is unencrypted. Clear-text communications are susceptible to sniffing, replay, and packet capture attacks. Another security risk is IP spoofing, where an attacker could impersonate a trusted IP address for the purpose of executing commands.

Most Junos device platforms do not have any remote access services enabled by default, but commands from this section can be used to disable insecure access services on deployed devices.

CAUTION Insecure access services could have been enabled on your production device for a number of reasons (transferring Junos images, troubleshooting, management, etc.). Be sure you research how they're used in your environment prior to disabling them so you don't impact operations.

How to Disable Insecure Access Services

1. Disable Berkeley "r" commands. These commands are used to provide different methods of convenient access without requiring a password. Both of these commands are undocumented and hidden but information is available from public sources on how to enable these services:

```
[edit system services]
jweidley@ex3200# delete rsh

[edit system services]
jweidley@ex3200# delete rlogin
```

2. Disable FTP:

```
[edit system services]
jweidley@ex3200# delete ftp
```

3. Disable Finger:

```
[edit system services]
jweidley@ex3200# delete finger
```

4. Disable Telnet:

```
[edit system services]
jweidley@ex3200# delete telnet
```

5. Disable J-Web via HTTP:

```
[edit system services]
jweidley@ex3200# delete web-management http
```

6. Disable Reverse Telnet:

```
[edit system services]
jweidley@ex3200# delete reverse telnet
```

7. Disable clear-text Junoscript access:

```
[edit system services]
jweidley@ex3200# delete xnm-clear-text
```

8. Disable TFTP server:

```
[edit system services]
jweidley@ex3200# delete tftp-server
```

NOTE If you attempt to disable an access service that is currently not enabled, Junos issues the following message:

```
[edit system services]
jweidley@ex3200# delete finger
warning: statement not found
```

This is a warning message and not an indication of a major problem – it's good infor- mation to know when you are applying your hardening template to multiple devices.

Enable Secure Access Services

Access services are considered secure when communication is encrypted and protected from snooping type attacks. This section shows how to enable them and explores a few options to further harden these secure services.

Secure Shell

Secure Shell (SSH) is a network protocol that allows data to be exchanged between devices using a secure channel. SSH was designed as a replacement to the Telnet and other insecure access protocols.

How to Secure SSH

1. Only use SSHv2 because there are inherent design flaws in SSHv1 which make it susceptible to man-in-the-middle attacks:

```
[edit system services]
jweidley@ex3200# set ssh protocol-version v2
```

2. Deny root user SSH access. When SSH is enabled all configured users will be able to access the device; this includes the default root account:

```
[edit system services]
jweidley@ex3200# set ssh root-login deny
```

Note that *not* disabling root SSH access presents two problems:

- A brute-force attack is where a malicious user attempts to guess usernames and passwords until they find the right combination that gives them access to the device. Every Junos device has a *root* account, so by not disabling remote root access you are providing the attackers with a known target account and all they have to do is guess the password.
- As previously discussed, the root account is associated with the default super-user login-class, which has idle-timeout disabled, so it will never be logged out due to inactivity.

3. Prevent users from creating an SSH tunnel over a CLI session to Junos device via SSH. This tunnel could be used for forward TCP traffic, bypassing firewall policies or ACLs, allowing access to resources beyond the router:

```
[edit system services]
jweidley@ex3200# set ssh no-tcp-forwarding
```

4. Limit the total number of unique connections to the device to preserve resources and reduce the chance of DoS. The connection limit varies on Junos platforms so research a value that works best for your devices:

```
[edit system services]
jweidley@ex3200# set ssh connection-limit 10
```

5. Limit the number of logins per minute to preserve resources and reduce the chance of a DoS attack. The rate limit varies on Junos platforms so research a value that works best for your devices:

```
[edit system services]
jweidley@ex3200# set ssh rate-limit 2
```

6. Modern SSH clients allow you to clone your existing session to another terminal, which is useful if you have to be logged in to the same device more than once for debugging and troubleshooting. Each cloned session is tunneled through the initial SSH session instead of a new process being spawned. Every SSH session consumes resources so you should consider limiting the number of sessions per connection. Note that this is different than what we configured in Step 4, `connection-limit` restricts the number of connections and `max-sessions-per-connection` limits tunneled sessions over the same connection:

```
[edit system services]
jweidley@ex3200# set ssh max-sessions-per-connection 2
```

7. To ensure unresponsive SSH clients don't consume value system resources, configure client keep alive messages. The Junos device will periodically request a response from the client. If the client doesn't respond within the configured threshold it is considered inactive and the connection will be terminated. Set the keep alive interval to 10 seconds and the max number of missed messages to three:

```
[edit system services]
jweidley@ex3200# set ssh client-alive-interval 10

[edit system services]
jweidley@ex3200# set ssh client-alive-count-max 3
```

8. Higher security environments may require the use of stronger encryption and data integrity algorithms. Ciphers are used for symmetric session encryption and Message Authentication Codes (MACs) are used for date integrity verification. Let's use the `ciphers` and `macs` options to enable only the FIPS approved algorithms:

```
[edit system services]
jweidley@ex3200# set ssh ciphers aes256-ctr

[edit system services]
jweidley@ex3200# set ssh ciphers aes256-cbc

[edit system services]
jweidley@ex3200# set ssh ciphers aes192-ctr

[edit system services]
jweidley@ex3200# set ssh ciphers aes192-cbc

[edit system services]
jweidley@ex3200# set ssh ciphers aes128-ctr

[edit system services]
jweidley@ex3200# set ssh ciphers aes128-cbc

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha2-512-96

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha2-512

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha2-256-96

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha2-256

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha1-96

[edit system services]
jweidley@ex3200# set ssh macs hmac-sha1
```

9. Review the configuration:

```
[edit system services]
jweidley@ex3200# show ssh
root-login deny;
no-tcp-forwarding;
protocol-version v2;
max-sessions-per-connection 2;
ciphers [ aes256-ctr aes256-cbc aes192-ctr aes192-cbc aes128-ctr aes128-cbc ];
macs [ hmac-sha2-512 hmac-sha2-256 hmac-sha1 hmac-sha1-96 ];
client-alive-count-max 3;
client-alive-interval 10;
connection-limit 10;
rate-limit 2;
```

10. Now that SSH is set up, the next step is to limit connections to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit SSH

```
set firewall family inet filter protect-re term allow-ssh from source-address <CLIENT>
set firewall family inet filter protect-re term allow-ssh from protocol tcp
set firewall family inet filter protect-re term allow-ssh from destination-port ssh
set firewall family inet filter protect-re term allow-ssh then accept
```

NOTE Some Junos platforms, particularly low end devices, have restrictions on the number of concurrent SSH connections. Be sure to check platform-specific documentation before making changes to these values. You should also research reasonable "connection-limit" and "rate-limit" values for your environment. Be sure to consider normal operations and emergency situations when determining the values for your network.

Securing J-Web via HTTPS

J-Web is the name for Juniper's web user interface that is used for configuring and monitoring a Junos device. Enabling HTTPS for J-Web provides confidentiality and integrity when connecting to the device.

NOTE Generating a certificate is beyond the scope of this book, so this procedure assumes that you already have a X.509 SSL certificate that is signed by a trusted authority.

How to Secure J-Web

1. Obtain a valid X.509 certificate that has been digitally signed by a trusted certificate authority (CA) and copy it to your home directory on the Junos device.

2. Load your certificate so Junos can use it:

```
[edit]
jweidley@ex3200# edit security certificates

[edit security certificates]
jweidley@ex3200# set local ex3200-ssl-cert load-key-file ?
Possible completions:
  <load-key-file>     File (URL) containing an SSL certificate and
                    private key in PEM format
  ex3200cert.pem      Size: 2278, Last changed: May 19 20:39:04

[edit security certificates]
jweidley@ex3200# set local ex3200-ssl-cert load-key-file ex3200cert.pem
```

3. Enable HTTPS and use the new X.509 certificate:

```
[edit security certificates]
jweidley@ex3200# top edit system services web-management

[edit system services web-management]
jweidley@ex3200# set https local-certificate ex3200-ssl-cert
```

4. Enabling a session's idle timeout is always a good security practice to reduce the chances of unattended sessions being used by unauthorized users:

```
[edit system services web-management]
jweidley@ex3200# set session idle-timeout 30
```

5. Imposing a limit on the number of J-Web sessions ensures responsible use by authorized users, conserves system resources, and reduces the possibility of a DoS attack:

```
[edit system services web-management]
jweidley@ex3200# set session session-limit 4
```

6. (Optional) If out-of-band management is not used, you can ensure that J-Web connections are only accepted over specific interfaces, such as your management network facing interface:

```
[edit system services web-management]
jweidley@ex3200# set https interface ge-0/0/0
```

7. Review the configuration:

```
[edit system services web-management]
jweidley@ex3200# show
https {
   local-certificate ex3200-ssl-cert;
   interface ge-0/0/0.0;
}
session {
   idle-timeout 30;
   session-limit 4;
}

[edit system services web-management]
jweidley@ex3200# top show security
certificates {
   llocal {
ex3200-ssl-cert {
"-----BEGIN RSA PRIVATE KEY----- [ removed ] -----END CERTIFICATE-----\n"; ## SECRET-DATA
      }
   }
}
```

8. Now that J-Web is set up, the next step is to limit connections to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit J-Web

```
set firewall family inet filter protect-re term allow-jweb from source-address <CLIENT>
set firewall family inet filter protect-re term allow-jweb from protocol tcp
set firewall family inet filter protect-re term allow-jweb from destination-port https
set firewall family inet filter protect-re term allow-jweb then accept
```

TIP It's a best security practice to use valid SSL certificates that are signed by a trusted certificate authority instead of self-signed certificates. Although self-signed certificates provide the required confidentiality, you will still be presented with a certificate validation error because the certificate path cannot be validated to the root, so it's essentially training you to ignore certificate warnings.

Securing XNM

Junos Script is an Extensible Markup Language (XML) Network Management API (Application Programming Interface) that custom built client applications use to request and change configuration information and monitor devices that run the Juniper Networks Junos Software.

NOTE Generating a certificate is beyond the scope of this book, so this procedure assumes that you already have a X.509 SSL certificate that is signed by a trusted authority.

How to Secure XNM

1. Obtain a valid X.509 certificate that has been digitally signed by a trusted CA and copy it to your home directory on the Junos device.

2. Load your certificate so Junos can use it:

```
[edit]
jweidley@ex3200# edit security certificates

[edit security certificates]
jweidley@ex3200# set local ex3200-ssl-cert load-key-file ? Possible completions:
<load-key-file>    File (URL) containing an SSL certificate and private key in PEM format
ex3200cert.pem Size: 2278, Last changed: May 19 20:39:04

[edit security certificates]
jweidley@ex3200# set local ex3200-ssl-cert load-key-file ex3200cert.pem
```

3. Enable SSL and use the new X.509 certificate:

```
[edit security certificates]
jweidley@ex3200# top edit system services xnm-ssl

[edit system services xnm-ssl]
jweidley@ex3200# set local-certificate ex3200cert.pem
```

4. By default, Junos supports a large number of simultaneous connections. Let' s make it specific to XNM connection and set the connection limit lower than the default to preserve system resources and reduce the possibly of DoS attack:

```
[edit system services xnm-ssl]
jweidley@ex3200# set connection-limit 10
```

5. By default, Junos supports a large number of connections per minute. Let's make it specific to XNM connections and set a rate limit lower than the default to preserve system resources and reduce the possibly of DoS attack:

```
[edit system services xnm-ssl]
jweidley@ex3200# set rate-limit 4
```

6. Review the configuration:

```
[edit system services xnm-ssl]
jweidley@ex3200# show
```

```
local-certificate ex3200cert.pem;
connection-limit 10;
rate-limit 4;
```

7. The next step is to limit access to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit encrypted XNM connections:

```
set firewall family inet filter protect-re term allow-xnm from source-address <CLIENT>
set firewall family inet filter protect-re term allow-xnm from protocol tcp
set firewall family inet filter protect-re term allow-xnm from destination-port 3220
set firewall family inet filter protect-re term allow-xnm then accept
```

Securing NETCONF

The NETCONF XML management protocol is a standards-based (RFC4741) XML-based protocol that client applications use to request and change configuration information on routing, switching, and security devices.

The NETCONF XML management protocol uses an XML-based data encoding for the configuration data and remote procedure calls. Client applications access the NETCONF server using the SSH protocol and use the standard SSH authentication mechanism. After authentication, the NETCONF server uses the configured Junos OS login usernames and classes to determine whether a client application is authorized to make each request.

1. By default, Junos supports a large number of simultaneous connections. Let's make is specific to NETCONF connections and set the connection limit lower than the default to preserve system resources and reduce the possibly of DoS attack:

```
[edit system services netconf]
jweidley@ex3200# set ssh connection-limit 10
```

2. By default, Junos supports a large number of connections per minute. Let's make is specific to NETCONF connections and set a rate limit lower than the default to preserve system resources and reduce the possibly of DoS attack:

```
[edit system services netconf]
jweidley@ex3200# set ssh rate-limit 4
```

3.Review the configuration:

```
[edit system services netconf]
jweidley@ex3200# show
ssh {
   connection-limit 10;
   rate-limit 4;
}
```

4. The next step is to limit NETCONF connections to trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

Sample firewall filter term to limit NETCONF:

```
set firewall family inet filter protect-re term allow-netconf from source-address <CLIENT>
set firewall family inet filter protect-re term allow-netconf from protocol tcp
set firewall family inet filter protect-re term allow-netconf from destination-port 830
set firewall family inet filter protect-re term allow-netconf then accept
```

User Authentication

Other important aspects of hardening your network devices are user authentication, command authorization, and permissions. All user access requests require some type of authentication. User authentication requests can be validated either locally or remotely via RADIUS or TACACS+.

In larger networks, centralized authentication is almost a necessity to consistently enforce password policies and manage user accounts. But whether you are in a large or small network, at least one local user account should always be configured on your network devices for emergency purposes.

The next section shows you how to configure user permissions using custom login classes, RADIUS Authentication & Accounting, TACACS+ Authentication & Accounting, and local user accounts, in addition to exploring the features necessary to implement a password complexity policy.

Login Permissions

The information security principle called Least Privilege states that users, processes, services, and devices should only be given privileges for or have access to resources that are consistent with their function or assigned duties.

In Junos software user privileges are defined in a *login class*. All users that log in to a Junos device must be assigned a *login class*. Login classes allow you to define the following:

- Access privileges when the user logs in to the device
- Commands and statements that the user can and cannot execute
- Other useful options such as time-based enforcement, idle time, and displaying system alarms on login

Junos software provides four built-in login classes:

Login Class	Permissions	Description
operator	Clear, network, reset, trace, view	Can perform all actions available with these specific permission bits in operational mode. Operator cannot display or change the configuration and cannot shut down or reboot the device.
read-only	View	Can perform all actions available with the view permission in operational mode.
super-user	All	Can perform any operations on the device.
unauthorized	None	Can log in to the device but cannot perform any operations except logout.

These built-in login classes are not configurable and most likely not suitable for your production environment. The next section will show an example of how the Junos OS provides extensive access controls for user management.

MORE? For additional information regarding login class configuration see the Junos technical documentation, *Junos OS Basic System Configuration Guide*, at https://www.juniper.net/techpubs/en_US/junos14.2/topics/concept/access-login-class-overview.html.

How to Create Custom Login Classes

A firewall analogy can be applied when developing a least privileges concept for your engineers. Start off with a default deny posture, then review the responsibilities for each role within your organization and permit access to the corresponding Junos CLI hierarchy.

Think about the functions of the engineers or groups that will need access to your devices. Login permissions requirements are likely to be site specific, but in this section let's use the standard Tier 1, 2, 3 engineer example, and show how to permit and deny specific functions to those tiers. Our example requirements will be:

- Tier 1: Should be able to run `show` commands for basic troubleshooting and be able to run network tools (ping, traceroute, etc.) to determine reachability. They should be able to see the configuration but SECRET-DATA should be obscured.
- Tier 2: Should have all the functionality of Tier1 engineers but be able to clear device counters and statistics and make interface and routing configuration changes.
- Tier 3: Should have the ability to perform every function on the device to support management as well as advanced troubleshooting tasks like low level debugging, code upgrades, etc.

Tier1

Let's create a new login class for our Tier 1 network operators with permissions only to do the tasks that they need to accomplish.

1. Create the login class name:

```
{master:0}[edit]
jweidley@EX4500# edit system login class tier1
```

2. Set the inactivity time to a value that is reasonable for your environment. After this period of time, users will be automatically logged off of the system. Note that users will receive warning messages alerting them that their session will be timed out:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set idle-timeout 10
```

3. Login tips are helpful for engineers that are new to Junos as well as providing helpful tips about commands:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set login-tip
```

4. It's a good practice to alert engineers to the current system alarms so they can be investigated:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set login-alarms
```

5. `Maintenance` permissions permit useful maintenance-related tasks like shutting down and rebooting the device:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set permissions maintenance
```

6. `View` permissions permit the use of `show` commands, which are helpful for seeing routing tables, spanning-tree, interface statistics, etc.:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set permissions view
```

7. `Network` permissions permit the use of ping, traceroute, telnet, and SSH for testing connectivity and routing:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set permissions network
```

8. `View-configuration` permissions allow the configuration file to be viewed. Note that the entire configuration can be viewed but `SECRET-DATA` will be obscured:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set permissions view-configuration
```

9. By enabling some permissions, Tier 1 users have access to commands that they shouldn't use. The `deny-commands` option allows you to use standard regular expressions to specifically define the commands that the Tier 1 user should not have access to:

- (`start *`) restricts all `start` commands, which can be used to gain access to the underlying Unix shell
- (`set cli idle-timeout`) restricts the ability to redefine the idle-timeout value of the login class on a per session basis:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# set deny-commands "(start *)|(set cli idle-timeout)|(request system
software)|(request system zeroize)|(request chassis)"
```

10. Review the configuration:

```
{master:0}[edit system login class tier1]
jweidley@EX4500# show
idle-timeout 10;
login-alarms;
login-tip;
permissions [ maintenance network view view-configuration ];
deny-commands "(start *)|(set cli idle-timeout)|(request system software)|(request system
zeroize)|(request chassis)";
```

MORE? For additional information and examples of regular expression matches see the Junos technical documentation, *Junos OS Access Privilege Administration Guide*, at http://www.juniper.net/techpubs/en_US/junos13.3/information-products/pathway-pages/access-privilege/access-privilege.html.

Tier 2

Now let's create another login class for the Tier 2 engineers with additional permissions to do more advanced functions.

1. Define the name of the login class:

```
{master:0}[edit system login]
jweidley@EX4500# edit class tier2
```

2. Set the inactivity time to a value that is reasonable for your environment. After this period of time, users will be automatically logged off of the system. Note that users will receive warning messages alerting them that their session will be timed out:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set idle-timeout 15
```

3. It's a good practice to alert engineers to the current system alarms so they can be investigated:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set login-alarms
```

4. Tier 2 engineers should have at least the same permissions as the Tier 1 engineers, so configure all of them with the same permissions:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions maintenance

{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions network

{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions view

{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions view-configuration
```

5. `Clear` permissions allow counters and statistics to be cleared, which is helpful in some troubleshooting scenarios:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions clear
```

6. `Configure` permissions allow users to enter configuration mode, but by default don't give permissions to actually configure anything useful. So let's allow Tier 2 engineers to configure interfaces and routing-related information to support provisioning:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions configure

{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions interface-control

{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions routing-control
```

7. `Rollback` permission allows configuration changes to be rolled back in the event of an error:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set permissions rollback
```

8. Use the `deny-commands` option to selectively restrict specific commands:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set deny-commands "(start *)|(set cli idle-timeout)|(request system
software)|(request system zeroize)"
```

9. Use the `deny-configuration` option to selectively restrict changing a subset of permitted sections of the configuration. Here we will restrict their ability to change the `groups` portion of the configuration:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# set deny-configuration "(groups)"
```

10. And let's review the configuration:

```
{master:0}[edit system login class tier2]
jweidley@EX4500# show
idle-timeout 15;
login-alarms;
permissions [ clear configure interface-control maintenance network rollback routing-control view
view-configuration ];
deny-commands "(start *)|(set cli idle-timeout)|(request system software)|(request system
zeroize)";
deny-configuration "(groups)";
```

Tier 3

For the Tier3 engineers you could just use the built-in super-user login class, but here let's create a new super-user class with an idle timeout and other customizations.

1. Create a custom class and define the name of the login class:

```
{master:0}[edit system login]
lab@EX4500# edit class tier3
```

2. Set the inactivity time to a value that is reasonable for your environment. After this period of time, users will be automatically logged off of the system. Note that users will receive warning messages alerting them that their session will be timed out:

```
{master:0}[edit system login class tier3]
lab@EX4500# set idle-timeout 20
```

3. It's a good practice to alert engineers to the current system alarms so they can be investigated:

```
{master:0}[edit system login class tier3]
jweidley@EX4500# set login-alarms
```

4. Use the `all` keyword to provide access to all permissions:

```
{master:0}[edit system login class tier3]
lab@EX4500# set permissions all
```

5. And let's verify the configuration:

```
{master:0}[edit system login class tier3]
lab@EX4500# show
idle-timeout 20;
login-alarms;
permissions all;
```

Login class configuration is specific to each organization and the Junos OS has many options available to help you design a policy to meet your needs. You will have to experiment with different configurations until you find something that meets your requirements.

TIP There are other options available under the `[system login class]`, such as access times and days of the week, that are helpful in enhancing your login policy. These are options you can use to meet your organization's security policy requirements, if necessary.

MORE? For an explanation of the different login class permissions visit http://www.juniper.net/techpubs/en_US/junos14.1/topics/concept/access-privileges-levels-overview.html.

MORE? For more login class examples see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006), at *http://www.juniper.net/books*.

Try It Yourself: Protecting ## SECRET-DATA

Sensitive authentication information is stored in the configuration and access should be restricted to small number of trusted engineers. Please refer to the *Password Storage and Password Access* sections in Chapter 3 for more details.

The `tier1` and `tier2` users in the above sample configurations are able to see the entire configuration but are not able to see the stored passwords. The stored authentication information is removed from view and replaced with `/* SECRET-DATA */`:

```
{primary:node0}
tier2@srx650> show configuration system ntp
boot-server 192.168.220.41;
authentication-key 1 type md5 value /* SECRET-DATA */; ## SECRET-DATA
server 192.168.220.41 key /* SECRET-DATA */; ## SECRET-DATA
server 192.168.221.41 key /* SECRET-DATA */; ## SECRET-DATA
trusted-key 1;
```

Now compare the output to the `tier3` user, who is allowed to see the `SECRET-DATA`:

```
{primary:node0}
tier3@srx650> show configuration system ntp
boot-server 192.168.220.41;
authentication-key 1 type md5 value "$9$lIBKLxws3JUjwsaUBi.mZs5oZjik.O468Ctu"; ## SECRET-DATA
server 192.168.220.41 key 1; ## SECRET-DATA
server 192.168.221.41 key 1; ## SECRET-DATA
trusted-key 1;
```

User permissions can be verified using the `show cli authorization` command. The `tier2` user output below shows clearly that they can see the configuration but not the secrets:

```
{primary:node0}
tier2@srx650> show cli authorization | match "Permission|secret" Permissions:
view-configuration-- Can view all configuration (not including secrets)

{primary:node0}
tier2@srx650>
```

The `show cli authorization` output for the `tier3` user can also see the configuration but has additional permissions to see and modify the secret data:

```
{primary:node0}
tier3@srx650> show cli
authorization | match “Permission|secret”
Permissions:
secret-- Can view secret statements
secret-control-- Can modify secret statements
view-configuration-- Can view all configuration (not including secrets)

{primary:node0}
tier3@srx650>
```

RADIUS and RADIUS Accounting

Remote Authentication Dial-in User Service (RADIUS) is an industry standard protocol that provides centralized authentication, authorization, and accounting (AAA) management for network devices.

The RADIUS protocol does not transmit passwords in plaintext over the network. Instead, it uses an agreed upon, shared, secret password and a hashing algorithm to secure passwords between the client and the server. Security is increased by segregating user and management data by using an out-of-band management network (an overly inquisitive user can't attempt to sniff, crack, or replay the RADIUS messages if they can't see them).

NOTE There is a degree of flexibility in the Junos OS RADIUS client implementation and RADIUS server software packages that is beyond the scope of this book. This section focuses on configuring secure communication and proper accounting configuration from a network device perspective.

MORE? For additional information regarding RADIUS configuration and vendor specific attributes (VSAs) see the Junos technical documentation, *Junos OS Basic System Configuration Guide*, at http://www.juniper.net/documentation/en_US/junos14.1/topics/reference/general/radius-vendor-specific-attributes-juniper-networks.html.

How to Configure RADIUS Authentication

1. First, let's define the IP address of the RADIUS server:

```
[edit]
jweidley@ex3200# edit system radius-server 192.168.3.20
```

2. Set the shared secret password (this should be a strong password that is difficult to guess and that follows your password complexity policy):

```
[edit system radius-server 192.168.3.20]
jweidley@ex3200# set secret $2rK-nh%Aj4WQ=}
```

3. Define the port the RADIUS server accepts requests on. UDP 1812 is the default port, but specifically defining it makes your configuration more precise:

```
[edit system radius-server 192.168.3.20]
jweidley@ex3200# set port 1812
```

4. (Optional) If `set system default-address-selection` isn't configured, set the source address to ensure RADIUS requests are sourced from a predictable address. It is common to use the IP address of the lo0 interface:

```
[edit system radius-server 192.168.3.20]
jweidley@ex3200# set source-address 192.168.70.1
```

5. Verify the configuration:

```
[edit system radius-server 192.168.3.20]
jweidley@ex3200# show
port 1812;
secret "$9$oIJjH.P5F69mPRhylMWjHkqQF"; ## SECRET-DATA
source-address 192.168.70.1;
```

The Junos RADIUS implementation supports password changes when accounts have expired, have been reset, or have been configured to have the password changed at next logon. This is done by configuring the Microsoft version of the Challenge Handshake Authentication Protocol (MS-CHAP):

```
[edit]
jweidley@ex3200# edit system radius-options

[edit system radius-options]
jweidley@ex3200# set password-protocol mschap-v2

[edit system radius-options]
jweidley@ex3200# show
password-protocol mschap-v2;
```

NOTE For the purpose of brevity, this example only shows the configuration of one RADIUS server. It is recommended that you configure multiple RADIUS servers on separate subnets for resiliency.

How to Configure RADIUS Accounting

1. Configure the events for which you want to receive accounting messages. In this example configuration let's configure all available options for full auditing capabilities:

```
[edit]
jweidley@ex3200# edit system accounting

[edit system accounting]
jweidley@ex3200# set events login

[edit system accounting]
jweidley@ex3200# set events change-log

[edit system accounting]
jweidley@ex3200# set events interactive-commands
```

2. Define the IP address of the RADIUS server and set the accounting port (UDP 1813 is the default RADIUS accounting port, but specifically configuring it makes your configuration more precise):

```
[edit system accounting]
jweidley@ex3200# edit destination radius server 192.168.3.20

[edit system accounting destination radius server 192.168.3.20]
jweidley@ex3200# set accounting-port 1813
```

3. Set the shared secret password. This should be a strong password that is difficult to guess and should follow your password complexity policy:

```
[edit system accounting destination radius server 192.168.3.20]
jweidley@ex3200# set secret M1zoVg2NRa8:r#r
```

4. (Optional) If `set system default-address-selection` isn't configured, set the source address to ensure RADIUS requests are sourced from a predictable address. It is common to use the IP address of the lo0 interface:

```
[edit system accounting destination radius server 192.168.3.20]
jweidley@ex3200# set source-address 192.168.70.1
```

5. Verify the configuration:

```
[edit system accounting destination radius server 192.168.3.20]
jweidley@ex3200# show
accounting-port 1813;
secret "$9$cxFyvWX7-w24x7k.fT3nvW8LVw"; ## SECRET-DATA
source-address 192.168.70.1;
```

6. The next step is to limit RADIUS connections from trusted sources. See the sample the firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

NOTE For the purpose of brevity, this example only shows the configuration of one RADIUS Accounting server. It is recommended that you configure multiple RADIUS servers on separate subnets for resiliency.

Sample firewall filter term to limit RADIUS

```
set firewall family inet filter protect-re term allow-radius from source-address <SERVER>
set firewall family inet filter protect-re term allow-radius from protocol udp
set firewall family inet filter protect-re term allow-radius from source-port radius
set firewall family inet filter protect-re term allow-radius from source-port radacct
set firewall family inet filter protect-re term allow-radius then accept
```

TACACS+ and TACACS+ Accounting

Terminal Access Controller Access Control System Plus (TACACS+) is the latest version of the older TACACS authentication software. TACACS+ is an AAA protocol used to provide access control to network devices.

NOTE There is a degree of flexibility in the Junos OS TACACS+ client implementation and TACACS+ server software packages that is beyond the scope of this book. This section focuses on configuring secure communication and proper accounting configuration from a network device perspective.

MORE? For additional information regarding TACACS+ configuration and vendor specific attributes (VSAs) see the Junos technical documentation, *Junos OS Basic System Configuration Guide*, at http://www.juniper.net/techpubs/en_US/junos14.1/topics/reference/general/tacacs-vendor-specific-attributes-juniper-networks.html.

How to Configure TACACS+ Authentication

1. Define the IP address of the TACACS+ server:

```
[edit]
```

```
jweidley@ex8208# edit system tacplus-server 192.168.3.40
```

2. Set the shared secret password. This should be a strong password that is difficult to guess and that follows your password complexity policy:

```
[edit system tacplus-server 192.168.3.40]
jweidley@ex8208# set secret $2rK-nh%Aj4WQ=}
```

3. Configure the default TACACS+ port to TCP 49. Although it is the default port, specifically configuring it makes your configuration more precise:

```
[edit system tacplus-server 192.168.3.40]
jweidley@ex8208# set port 49
```

4. (Optional) If `set system default-address-selection` isn't configured, set the source address to ensure TACACS+ requests are sourced from a predictable address. It is common to use the IP address of the lo0 interface:

```
[edit system tacplus-server 192.168.3.40]
jweidley@ex8208# set source-address 192.168.70.1
```

5. Verify the configuration:

```
[edit system tacplus-server 192.168.3.40]
jweidley@ex8208# show
port 49;
secret "$9$RywhlKWLxdwY8LjH.PQzlKvMNd"; ## SECRET-DATA
source-address 192.168.70.1;
```

NOTE For the purpose of brevity, this example only shows the configuration of one TACACS+ server. It is recommended that you configure multiple TACACS+ servers on separate subnets for resiliency.

How to Configure TACACS+ Accounting

1. Set the events that you want to track via accounting. In this example configuration let's configure all available options for full auditing capabilities:

```
[edit system tacplus-server 192.168.3.40]
jweidley@ex8208# top edit system accounting

[edit system accounting]
jweidley@ex8208# set events login

[edit system accounting]
jweidley@ex8208# set events change-log

[edit system accounting]
jweidley@ex8208# set events interactive-commands
```

2. Define the IP address of the TACACS+ Accounting server:

```
[edit system accounting]
jweidley@ex8208# edit destination tacplus server 192.168.3.40
```

3. Set the shared secret password. This should be a strong password that is difficult to guess and that follows your password complexity policy:

```
[edit system accounting destination tacplus server 192.168.3.40]
jweidley@ex8208# set secret M1zoVg2NRa8:r#r
```

4. Configure the default TACACS+ accounting port to TCP 49. Although it is the default port, specifically configuring it makes your configuration more precise:

```
[edit system accounting destination tacplus server 192.168.3.40]
jweidley@ex8208# set port 49
```

5. (Optional) If `set system default-address-selection` isn't configured, set the source address to ensure TACACS+ requests are sourced from a predictable address. It is common to use the IP address of the lo0 interface:

```
[edit system accounting destination tacplus server 192.168.3.40]
jweidley@ex8208# set source-address 192.168.70.1
```

6. Verify the configuration:

```
[edit system accounting destination tacplus server 192.168.3.40]
jweidley@ex8208# show
port 49;
secret "$9$ZsUk.fTz6Ct5TcyevLXk.mP36"; ## SECRET-DATA
source-address 192.168.70.1;
```

7. The next step is to limit TACACS connections from trusted sources. See the sample firewall filter below and the section in this chapter on *Protecting the Routing Engine* for more details.

NOTE For the purpose of brevity, this example only shows the configuration of one TACACS+ Accounting server. It is recommended that you configure multiple TACACS+ servers on separate subnets for resiliency.

Sample firewall filter term to limit TACACS+

```
set firewall family inet filter protect-re term allow-tacacs from source-address <SERVER>
set firewall family inet filter protect-re term allow-tacacs from protocol tcp
set firewall family inet filter protect-re term allow-tacacs from source-port 49
set firewall family inet filter protect-re term allow-tacacs then accept
```

Password Complexity

When configuring local accounts, you should ensure engineers choose passwords of an acceptable strength so your network devices are not compromised by something as simple as a poorly chosen password. Newer releases of Junos have introduced new features that provide a lot more granularity and flexibility that allows for the creation of a strong local password policy.

Creating a Password Policy

This book makes many references to a password policy and also implies that it should clearly define the criteria for what your organization considers an "acceptable" password. This section creates a password policy that will be utilized to ensure that local user account passwords meet defined security requirements.

For this example, our sample password policy will be that passwords should:

- use SHA1
- be a minimum of 15 characters in length
- contain at least two numbers

- contain at least two upper case characters
- contain at least two lower case characters
- contain at least two special characters

1. Set the minimum password length criteria within a range of 6 - 20 characters:

```
[edit]
jweidley@ex4200# edit system login password

[edit system login password]
jweidley@ex4200# set minimum-length 15
```

2. Define the `change-type` as `character-sets` so Junos will track the total number of character sets (uppercase, lowercase, numbers, and special characters) used:

```
[edit system login password]
jweidley@ex4200# set change-type character-sets
```

3. When using `change-type character-sets`, the number of character sets included in the password are checked against the specified minimum value configured:

```
[edit system login password]
jweidley@ex4200# set minimum-changes 4
```

4. Set the minimum number of numeric characters required:

```
[edit system login password]
jweidley@ex4200# set minimum-numerics 2
```

5. Set the minimum number of uppercase and lowercase characters required:

```
[edit system login password]
jweidley@ex4200# set minimum-upper-cases 2

[edit system login password]
jweidley@ex4200# set minimum-lower-cases 2
```

6. Set the minimum number of punctuation characters required:

```
[edit system login password]
jweidley@ex4200# set minimum-punctuations 2
```

7. Password security will be enhanced by using SHA1. Note that starting with Junos 13.3 you have the option of using the SHA256 and SHA512 cryptographic algorithms for increased security:

```
[edit system login password]
jweidley@ex4200# set format sha1
```

8. Review the configuration:

```
[edit system login password]
jweidley@ex4200# show
minimum-length 15;
change-type character-sets;
minimum-changes 4;
minimum-numerics 2;
minimum-upper-cases 2;
minimum-lower-cases 2;
minimum-punctuations 2;
format sha1;
```

MORE? For a detailed description of *change-types* and *set-transitions* to determine which is best for your environment, see the Junos technical documentation, *Junos OS Basic System Configuration Guide*, at http://www.juniper.net/techpubs/en_US/junos14.1/topics/concept/authentication-plain-text-password-requirements.html.

CAUTION User accounts created prior to configuring password complexity options will be unaffected by the new password requirements and should have their passwords reset.

Local Login Accounts and Template Accounts

Like most network operating systems, the Junos OS allows for the creation of local user accounts, and there are two types of local user accounts on a Junos device: the root user and a regular user account. As described in Chapter 3, the root account is the only default account and it is the most powerful account in the system. This section focuses on regular user accounts.

Junos can also use template accounts to assign permissions for users that authenticate via RADIUS and/or TACACS+. Each template account can define a different set of permissions appropriate for a group of users who use that template. These templates are defined locally on the router and referenced by the TACACS+ and RADIUS authentication servers via Vendor Specific Attributes (VSAs).

MORE? For additional information regarding local user accounts and template accounts, see the Junos technical documentation, *Junos OS User Access and Authentication Feature Guide for Routing Devices,* at http://www.juniper.net/techpubs/en_US/junos14.1/information-products/pathway-pages/system-basics/user-access.pdf.

How to Create Local User Accounts

Even if you're using centralized authentication, it is recommended that you always have at least one local account configured on the system, because local accounts provide an alternative way to log in to the device if RADIUS/TACACS+ is unavailable.

1. Define the local user name, that here is called *emergency*. Let's use the `full-name` option to document the purpose of this account:

```
[edit system login]
admin@j6350# set user emergency full-name "Emergency Only Local Account"
```

2. Assign the level of permissions the user should have on the device by assigning the appropriate default or custom `login class`. Let's use the `tier3` class that was created in the previous section, which provides full access and an idle-timeout:

```
[edit system login]
admin@j6350# set user emergency class tier3
```

3. Optionally configure a User ID (`uid`) that is consistent across Junos devices and UNIX systems to eliminate possible file ownership issues when transferring files (if a user ID isn't specified, Junos will automatically assign one):

```
[edit system login]
admin@j6350# set user emergency uid 2010
```

4. Finally, assign the password for the account:

```
[edit system login]
admin@j6350# set user emergency authentication plain-text-password
```

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

TIP The `full-name` option can also be used to provide contact information for that user account such as phone number, department the engineer works in, etc.

How to Create Template Accounts

Let's create template accounts for our engineers who will obtain their permissions from the login classes that were configured in the *Login Permissions* section of Chapter 3.

1. Define the local user name and you can use the `full-name` option to provide a description of the purpose of the account:

```
[edit system login]
admin@j6350# set user tier1 full-name "Login template for Tier1 Users"
```

2. Assign the level of permissions the user should have on the device by assigning the appropriate default or custom login class. Let's use the classes created in the previous section:

```
[edit system login]
admin@j6350# set user tier1 class tier1
```

3. Optionally configure a user ID (`uid`) that is consistent across Junos devices and UNIX systems to eliminate possible file ownership issues when transferring files (if a user ID isn't specified, Junos will automatically assign one):

```
[edit system login]
admin@j6350# set user tier1 uid 2001
```

4. To finish up the configuration, let's add the commands to add the Tier2 and Tier3 templates:

```
[edit system login]
admin@j6350# set user tier2 full-name "Login template for Tier2 Users"

[edit system login]
admin@j6350# set user tier2 class tier2

[edit system login]
admin@j6350# set user tier2 uid 2002

[edit system login]
admin@j6350# set user tier3 full-name "Login template for Tier3 Users"

[edit system login]
admin@j6350# set user tier3 class tier3

[edit system login]
admin@j6350# set user tier3 uid 2003
```

5. Review the configuration:

```
[edit system login]
user tier1 {
    full-name "Login template for Tier1 Users";
    uid 2001;
    class tier1;
}
user tier2 {
    full-name "Login template for Tier2 Users";
```

```
    uid 2002;
    class tier2;
}
user tier3 {
    full-name "Login template for Tier3 Users";
    uid 2003;
    class tier3;
}
```

TIP The configuration for creating a user template is very similar to a login account with the exception of setting a password. The password is not required because password validation is handled by the external authentication server (RADIUS/TACACS+).

Try it Yourself: Verifying Accountability

Security auditors frown on group accounts because you lose accountability from an auditing perspective. When using centralized authentication, it is important to understand that template accounts still provide a granular accountability of user logins and commands executed.

With RADIUS authentication configured and users being mapped to a user template, let's see how to verify permissions, verify that the template account is applied, and verify user accountability. User authorization information can be displayed with the `show cli authorization` command:

```
jweidley@mx960> show cli authorization
Current user: tier3 login: jweidley class tier3
Permissions:
  admin -- Can view user accounts
  admin-control-- Can modify user accounts
  clear -- Can clear learned network information
  configure  -- Can enter configuration mode
  control -- Can modify any configuration
```

In the first line of the output, the current user is the template account name (`tier3`). The login field is the actual username that authenticated to the device (jweidley), and class is the login class that was assigned by the login template (**`tier3`**).

If you review the interactive-commands log file, you can see that Junos tags all logs with the user's login name, not the login template name, to maintain full accountability:

```
Jun 15 14:08:54.439 2011 mx960-1 mgd[88155]: %INTERACT-6-UI_CMDLINE_READ_LINE: User jweidley,
command show configuration
Jun 15 14:09:18.783 2011 mx960-1 mgd[88155]: %INTERACT-6-UI_CMDLINE_READ_LINE: User jweidley,
command show log messages
Jun 15 14:09:36.352 2011 mx960-1 mgd[88155]: %INTERACT-6-UI_CMDLINE_READ_LINE: User jweidley,
command show log security
```

Limiting Login Attempts

A password guessing attack is a method to gain unauthorized access to a device by making repeated guesses at a user's password. The probability of success can be statistically reduced by defining and enforcing more stringent threshold limits.

How to Hinder Password Guessing Attacks for Local Accounts

The Junos default behavior for login security provides reasonable protection from password guessing attacks, but may not be suitable for every environment. Let's explore some options that are available to strengthen protection from unauthorized login access attempts.

1. Limit the maximum number of times a user is allowed to attempt to authenticate with the wrong password before the connection is terminated. The range is from 1 through 10. The system default is 10:

```
{master:0}[edit system login retry-options]
jweidley@EX4500# set tries-before-disconnect 3
```

2. Set the threshold for the number of failed login attempts before the user experiences a delay between login attempts. The range is from 1 through 3, with a default of 2:

```
{master:0}[edit system login retry-options]
jweidley@EX4500# set backoff-threshold 1
```

3. Define the delay time after each failed login attempt in seconds. The delay increases by this value for each subsequent login attempt after the value specified in the `backoff-threshold` option. The range is from 5 through 10. The default is 5:

```
{master:0}[edit system login retry-options]
jweidley@EX4500# set backoff-factor 6
```

4. Limit the minimum length of time in seconds that the connection remains open while the user is attempting to enter a password to log in. The range is from 20 through 60. The default setting is 20:

```
{master:0}[edit system login retry-options]
jweidley@EX4500# set minimum-time 30
```

5. Limit the maximum length of time in seconds available for a user to enter a username and password before the connection is terminated. The range is from 20 through 30 with a default of 120:

```
{master:0}[edit system login retry options]
jweidley@EX4500# set maximum-time 60
```

6. Set the amount of time in minutes before a user can attempt to login after being locked out due to the number of failed login attempts specified in Step 1 with the tries-before-disconnect statement. The range is from 1 through 43200 with a default of 120:

```
{master:0}[edit system login retry options]
jweidley@EX4500# set lockout-period 10
```

7. And let's verify the configuration:

```
{master:0}[edit system login retry-options]
jweidley@EX4500# show
tries-before-disconnect 3;
backoff-threshold 1;
backoff-factor 6;
minimum-time 30;
maximum-time 60;
lockout-period 10;
```

TIP You can also use the `show system login lockout` command to view whether any users are currently locked out, and when the lockout period begins and ends for each user. After a user is locked out of the device, the security administrator can use the `clear system login lockout username` command to manually remove the lockout for a user.

Authentication Order

Junos supports multiple authentication methods for user logins. There is also flexibility to configure multiple authentication methods. The `authentication-order` option is used to prioritize the order in which Junos utilizes those configured authentication methods.

This section reviews the subtle concepts to consider when setting the authentication order. Whether or not the local password database is specified in the `authentication-order` statement, it is always an option, and the key here is to understand when it will be used.

For each login attempt, the Junos OS tries the configured authentication methods in a set order until the password is accepted. If the username and password are accepted, the login attempt succeeds and no other authentication methods are tried. The next method is tried if the previous authentication method fails to respond, or if the method returns a reject response to the login attempt because of an incorrect username or password.

If none of the configured authentication methods accept the login credentials, and if a reject response is received, the login attempt fails. If no response is received from any configured authentication method, the Junos software consults local password authentication as a last resort.

The `authentication-order` option works the same for all authentication methods so for this example let's use RADIUS:

```
[edit system]
jweidley@ex4200# show authentication-order
authentication-order [ radius password ];
```

The RADIUS server will be consulted when users attempt to authenticate. If the RADIUS server returns a reject because the password is mistyped, or the account is not configured, the local password database is then consulted. This means that if you have local accounts configured, like the emergency account, that are not in RADIUS, users will be able to log in with that account even if the RADIUS servers are available.

If you want to always use RADIUS for user authentication and only consult the local password database if the RADIUS server is unreachable, consider the following configuration:

```
[edit system]
jweidley@ex4200# show authentication-order
authentication-order radius;
```

This example is the preferred choice in most situations because it increases security by capitalizing on the benefits of using centralized authentication but still provides a contingency if the servers are unavailable.

MORE? For more about the authentication ordering in Junos see *Junos Enterprise Routing, 2nd Edition* (by Southwick, Marschke & Reynolds, O'Reilly Media, 2011) at http://www.juniper.net/books.

Network Resiliency Protocol Authentication

In this section we'll configure authentication for two network protocols that increase the availability and resiliency of the network. Those protocols are VRRP (Virtual Router Redundancy Protocol) and BFD (Bidirectional Forwarding Detection).

VRRP Authentication

VRRP enables hosts on a LAN to make use of redundant routing platforms on that LAN without requiring more than the static configuration of a single default route on the hosts. The VRRP routing platforms share the virtual IP address corresponding to the default route configured on the hosts. At any time, one of the VRRP routing platforms is the master (active) and the others are backups. Routers running VRRP dynamically elect master and backup routers through the use of priorities. If authentication is not configured, it is possible for a malicious device to disrupt the election process and cause a denial of service attack.

By default VRRP authentication is disabled. Junos allows for two authentication options: simple and MD5. This section will configure MD5 authentication:

1. Start with a working VRRP configuration on the interface:

```
[edit interfaces ge-0/0/0 unit 0 family inet]
jweidley@mx80# show
address 192.168.5.3/24 {
   vrrp-group 1 {
       virtual-address 192.168.5.1;
       priority 100;
   }
}
```

2. Set the authentication type to MD5:

```
[edit interfaces ge-0/0/0 unit 0 family inet]
jweidley@mx80# set address 192.168.5.1/24 vrrp-group 1 authentication-type md5
```

3. Set the authentication key in accordance with your organization's password policy:

```
[edit interfaces ge-0/0/0 unit 0 family inet]
jweidley@mx80# set address 192.168.5.1/24 vrrp-group 1 authentication-key 8Xx-k.OIcyMW4
```

4. Review the configuration:

```
[edit interfaces ge-0/0/0 unit 0 family inet]
jweidley@mx80# show
address 192.168.5.3/24 {
   vrrp-group 1 {
       virtual-address 192.168.5.1;
       priority 100;
       authentication-type md5;
       authentication-key "$9$t3RApRhM8x7VYrl-w2aGU3n/CpOREylMXoJCpuORE"; ## SECRET-DATA
   }
}
```

5. Make the same configuration on the connected device using the same authentication type and key.

CAUTION When firewall filters are used to protect the routing engine, ensure that both IP Protocol 112 (VRRP) and IP protocol 51 (Authentication Header) are permitted.

See the section in this chapter on *Protecting the Routing Engine* and use this sample firewall filter term to limit VRRP.

Set the firewall filter term to limit VRRP

```
set firewall family inet filter protect-re term allow-vrrp from destination-address 224.0.0.18/32
set firewall family inet filter protect-re term allow-vrrp from protocol vrrp
set firewall family inet filter protect-re term allow-vrrp from protocol ah
set firewall family inet filter protect-re term allow-vrrp then accept
```

BFD Authentication

The BFD protocol is a simple mechanism that detects failures in a network and works in a wide variety of network environments and topologies. In BFD operation, devices exchange BFD hello packets at a specified interval and detect a neighbor failure if they do not receive a reply after a specified interval.

BFD neighbors can be directly connected or multiple hops away. As the number of hops increases so does the probability of exposure to attack. If BFD authentication is not configured, it is possible for a malicious user or device to cause a DoS attack through the use of spoofing or replay attacks.

BFD configuration is done in conjunction with the configuration of other routing protocols. Instead of providing configuration examples for all protocols that support BFD, in this section we will discuss the implementation at a high level and then configure BFD for IS-IS. There are three steps to configuring BFD Authentication:

1. Specify the BFD authentication algorithm for the protocol, IS-IS in this case:

```
[edit protocols isis interface ge-0/0/1.0]
jweidley@mx5# set bfd-liveness-detection authentication algorithm keyed-sha-1
```

2. Associate the authentication keychain with the protocol:

```
[edit protocols isis interface ge-0/0/1.0]
jweidley@mx5# set bfd-liveness-detection authentication key-chain bfd-isis-core
```

3. Configure the related security authentication keychain:

```
[edit protocols isis interface ge-0/0/1.0]
jweidley@mx5# top edit authentication-key-chains key-chain bfd-isis-core

[edit security authentication-key-chains key-chain bfd-isis-core]
jweidley@mx5# set key 1 secret D5vw~\H,[bI0aG4 start-time 2015-07-02.00:00
```

CAUTION Choose your authentication algorithms carefully, especially if you are using nonstop active routing (NSR). NSR enables a routing platform with redundant Routing Engines to switch from a primary Routing Engine to a backup Routing Engine without alerting peer nodes that a change has occurred. The `meticulous-keyed-md5` and `meticulous-keyed-sha-1` authentication algorithms are not supported when NSR is configured.

TIP When configured, strict authentication is enabled and authentication is checked at both ends of a BFD session. To reduce the chance of disruption when implementing BFD in an operational network, BFD has a `loose-check` option that accepts packets without strict authentication. The `loose-check` option is intended as a temporary transition mechanism and not for long-term use.

See the section in this chapter on *Protecting the Routing Engine* and use this sample firewall filter term to limit BFD:

Sample firewall filter term to limit BFD

```
set firewall family inet filter protect-re term allow-bfd from source-address <bfd neighbor>
set firewall family inet filter protect-re term allow-bfd from protocol udp
set firewall family inet filter protect-re term allow-bfd from source-port 49152-65535
set firewall family inet filter protect-re term allow-bfd from destination-port 3784-3785
set firewall family inet filter protect-re term allow-bfd from destination-port 4784
set firewall family inet filter protect-re term allow-bfd then accept
```

Routing Protocols and Route Authentication

There are many best practices related to route filtering and routing policies that contribute to stability and security but unfortunately they are beyond the scope of this book. So, this section of our short *This Week* book focuses only on authenticating routing updates with trusted neighbors.

Stable routing is very important to the overall success of any network. It is a recommended security practice to authenticate all routing protocol traffic to ensure that only trusted routers are participating in routing exchanges.

The threat is that an unauthorized router could send bogus routing advertisements to your router in an attempt to change or disrupt the normal flow of traffic. Traffic diversion can be used by malicious users or organizations in order to analyze it, or to cause a DoS attack.

To be effective, route authentication needs to be configured across the entire routing domain with every peer router. Even though the authentication keys are stored in a cryptographically obscured form in the Junos configuration, steps should be taken to restrict which engineers can view them.

CAUTION If you are running routing protocols through a stateful firewall and want to enable route authentication, take a look at your firewall vendor documentation to see how it supports TCP Sequence Number Randomization. This is important. If the firewall is randomizing sequence numbers it will cause the cryptographic checksums to be different and you won't be able to establish an adjacency.

BEST PRACTICE It is a best practice to use firewall filters to restrict routing protocol updates to trusted sources. See the sample firewall filters in each section and the section on *Protecting the Routing Engine* for more details.

RIP Authentication

In order to implement route authentication you must use RIP version 2. (RIPv2). RIPv2 supports two types of route authentication methods: simple and MD5. Simple route authentication provides minimal security but the "secret" key can be recovered by sniffing the protocol updates, which of course reduces its effectiveness. So let's focus on configuring MD5 as the authentication type.

MD5 is a widely used cryptographic hashing algorithm that generates a 128-bit hash value. The sending router inserts the configured MD5 hash in all transmitted RIP packets to its neighbors. When the receiving router receives the RIP packet it verifies this checksum before processing the packet contents.

One consideration is whether to enable authentication globally for all RIP groups or at the RIP group level, so let's configure it both ways.

How to Enable RIP Route Authentication for All Groups

This example configures route authentication for all RIP groups. Enabling authentication globally is recommended if you're only communicating with devices under your direct control, and it reduces configuration by only specifying the key once.

1. Start off with a simple RIP configuration:

```
[edit protocols rip]
jweidley@mx80# show
group eng-group {
   export advertise-static;
   neighbor ge-0/0/1.0;
}
```

2. Set the authentication type to MD5:

```
[edit protocols rip]
jweidley@mx80# set authentication-type md5
```

3. Set the authentication key. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols rip]
jweidley@mx80# set authentication-key D5vw~\H,[bI0aG4
```

4. And verify the configuration:

```
[edit protocols rip]
jweidley@mx80# show
authentication-type md5;
authentication-key "$9$XVy-VY4aUH.PaJT3n/puO1RhK8"; ## SECRET-DATA
group eng-group {
   export advertise-static;
   neighbor ge-0/0/1.0;
}
```

How to Enable RIP Route Authentication for Individual Groups

The authentication type and key can also be set at the group level. This provides the flexibility of using different authentication keys for different organizations or departments without sharing the authentication key values.

1. Start off with a simple RIP configuration:

```
[edit protocols rip]
jweidley@mx80# show
group eng-group {
   export advertise-static;
   neighbor ge-0/0/1.0;
}
```

2. At the group level, set the authentication type to MD5:

```
[edit protocols rip]
jweidley@mx80# edit group eng-group

[edit protocols rip group eng-group]
```

```
jweidley@mx80# set neighbor ge-0/0/1.0 authentication-type md5
```

3. Set the authentication key. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols rip group eng-group]
jweidley@mx80# set neighbor ge-0/0/1.0 authentication-key D5vw~\H,[bIOaG4
```

4. Verify the configuration:

```
[edit protocols rip group eng-group]
jweidley@mx80# up

[edit protocols rip]
jweidley@mx80# show
group eng-group {
   export advertise-static;
   neighbor ge-0/0/1.0 {
      authentication-type md5;
      authentication-key "$9$sv4aUikPQ36kqCu0BhcylKMNb"; ## SECRET-DATA
   }
}
```

MORE? For more detailed information regarding RIP see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

TIP Some of the reserved fields in RIP version 1 packets must be zero, whereas in RIP version 2 packets, most of these reserved fields can contain nonzero values. By default, RIP discards version 1 packets that have nonzero values in the reserved fields and version 2 packets that have nonzero values in the fields that must be zero. This default behavior implements the RIP version 1 and version 2 specifications but the `check-zero` option can be added to your configuration to document the default behavior.

See the section in this chapter on *Protecting the Routing Engine* and use the following sample firewall filter term to limit RIP:

Sample firewall filter term to limit RIP

```
set firewall family inet filter protect-re term allow-rip from source-address <NEIGHBOR>
set firewall family inet filter protect-re term allow-rip from destination-address 224.0.0.9/32
set firewall family inet filter protect-re term allow-rip from protocol udp
set firewall family inet filter protect-re term allow-rip from destination-port rip
set firewall family inet filter protect-re term allow-rip then accept
```

OSPF Route Authentication

OSPFv2 protocol exchanges can be authenticated to guarantee that only trusted routers can participate in the autonomous system's (AS) routing. Because OSPF performs authentication at the area level, all routers within the area must have the same authentication and corresponding password (key) configured.

Like RIP, OSPF also supports both simple and MD5 route authentication types. Simple route authentication does provide minimal security but the "secret" key can be recovered by sniffing the protocol updates, which reduces its effectiveness just as it did for RIP. So let's focus on configuring MD5 as the authentication type. When MD5 is enabled on an interface, that interface accepts routing updates only

if MD5 authentication succeeds. Otherwise, updates are rejected. The routing device only accepts OSPFv2 packets sent using the same key identifier (ID) that is defined for that interface.

MD5 is a widely used cryptographic hashing algorithm that generates a 128-bit hash value. The sending router inserts the configured MD5 hash in all transmitted OSPF packets to its neighbors. When the receiving router receives the OSPF packet it verifies this checksum before processing the packet contents.

How to Enable OSPF Authentication

1. Start with a basic OSPF configuration:

```
[edit protocols ospf]
jweidley@mx240# show
export advertise-static;
area 0.0.0.0 {
   interface ge-0/0/1.0;
}
```

2. Set the authentication type to MD5. Select a key ID between 0 and 255 – here let's use 1. Then set the key value, or password. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols ospf]
jweidley@mx240# edit area 0.0.0.0

[edit protocols ospf area 0.0.0.0]
jweidley@mx240# set interface ge-0/0/1.0 authentication md5 1 key D5vw~\H,[bIOaG4
```

3. Verify the configuration:

```
[edit protocols ospf]
jweidley@mx240# show
export advertise-static;
area 0.0.0.0 {
   interface ge-0/0/1.0 {
      authentication {
         md5 1 key "$9$N4bs4JGi.fTGUF"; ## SECRET-DATA
      }
   }
}
```

How to Enable OSPF Authentication with Automatic Key Rollover

It's a good security practice to periodically change statically configured passwords. This includes route authentication keys. This task becomes more tedious or possibly unmanageable in large networks.

This section will configure multiple MD5 keys, each with a unique key ID, and set the date and time to switch to a new key. The receiver of the OSPF packet uses the ID to determine which key to use for authentication.

1. Building from the previous example, configure a second authentication key and set the date and time it will start to be used:

```
[edit protocols ospf area 0.0.0.0]
jweidley@mx240# set interface ge-0/0/1.0 authentication md5 2 key 4}QYWwR+^@V7^uf start-time 2011-03-
31.16:32
```

2. And verify the configuration:

```
[edit protocols ospf area 0.0.0.0]
jweidley@mx240# show
interface ge-0/0/1.0 {
   authentication {
      md5 1 key "$9$N4bs4JGi.fTGUF"; ## SECRET-DATA
      md5 2 key "$9$v4DM7Vg4ZjkPJG" start-time "2011-3-31.16:32:00 -0700"; ## SECRET-DATA
   }
}
```

NOTE Since the key rotation is based on date and time it is imperative to have a stable and reliable NTP time source configured.

MORE? For more detailed information regarding OSPF see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

See the section in this chapter on *Protecting the Routing Engine* and use the following sample firewall filter term to limit OSPF:

Sample firewall filter term to limit OSPF

```
set firewall family inet filter protect-re term allow-ospf from destination-address <LOCAL IPs>
set firewall family inet filter protect-re term allow-ospf from destination-address 224.0.0.5/32
set firewall family inet filter protect-re term allow-ospf from destination-address 224.0.0.6/32
set firewall family inet filter protect-re term allow-ospf from protocol ospf
set firewall family inet filter protect-re term allow-ospf then accept
```

OSPFv3 Route Authentication

OSPFv3 is the version of OSPF that supports exchanging routing updates in IPv6 networks. OSPFv3 does not have a built-in authentication method and relies on the IP Security (IPsec) suite to provide this functionality. IPsec provides such functionality as authentication of origin, data integrity, confidentiality, replay protection, and nonrepudiation of source. You can use IPsec to secure specific OSPFv3 interfaces and protect OSPFv3 virtual links.

IPsec is based on security associations (SAs). Manual SAs require no negotiation between the peers. All values, including the keys, are static and specified in the configuration. Manual SAs statically define the Security Parameter Index (SPI) values, algorithms, and keys to be used and require matching configurations on both end points (OSPFv3 peers). As a result, each peer must have the same configured options for communication to be successful.

How to Configure OSPFv3 Route Authentication

This example configures OSPFv3 and the IPsec SA for security. Junos supports different hashing and encryption algorithms, and this example uses MD5, but you should choose the algorithm that best suits the security requirements of your network.

1. Start with a simple working OSPFv3 configuration:

```
[edit protocols ospf3]
jweidley@MX480# show
area 0.0.0.0 {
   interface lo0.0 {
      passive;
   }
   interface ge-0/0/1;
}
```

2. Configure the IPsec security association. First we'll use the optional `description` command to document the purpose of this SA. The mode will be set to `transport` which encrypts the data portion of the packet but leaves the header in clear-text. When using `transport` mode the protocol must be set to `ah` (authentication header). Finally, we'll configure a `bidirectional spi` (Security Parameter Index) with an arbitrary value between 256 - 16639.

```
[edit]
jweidley@MX480# edit security ipsec security-association ospf3-auth-core

[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set description ospf3-neighbor-auth-core

[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set mode transport

[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set manual direction bidirectional protocol ah

[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set manual direction bidirectional spi 256
```

3. Set the authentication algorithm for your environment, here using MD5:

```
[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set manual direction bidirectional authentication algorithm hmac-md5-96
```

4. Then set the key value, or password. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# set manual direction bidirectional authentication key ascii-text D5vw~\H,[bI0aG4j
```

5. Configure OSPFv3 to use the IPSEC SA for authentication:

```
[edit security ipsec security-association ospf3-auth-core]
jweidley@MX480# top edit protocols ospf3 area 0.0.0.0

[edit protocols ospf3 area 0.0.0.0]
jweidley@MX480# set interface ge-0/0/1 ipsec-sa ospf3-auth-core
```

6. And verify the configuration:

```
[edit protocols ospf3 area 0.0.0.0]
jweidley@MX480# up

[edit protocols ospf3]
jweidley@MX480# show
area 0.0.0.0 {
    interface lo0.0 {
        passive;
    }
    interface ge-0/0/1 {
        ipsec-sa ospf3-auth-core;
    }
}

[edit protocols ospf3]
jweidley@MX480# top show security ipsec security-association ospf3-auth-core
description ospf3-neighbor-auth-core;
mode transport;
manual {
    direction bidirectional {
        protocol ah;
        spi 256;
        authentication {
```

```
            algorithm hmac-md5-96;
            key ascii-text "$9$W5I8XNs24DHmGDEyrvx72goaUjz3/AtOreYgoaiHTQF3ApSyK"; ## SECRET-DATA
        }
    }
}
```

MORE? For more Junos IPv6 specific information, check out *Day One: Exploring IPv6* and *Day One: Advanced IPv6 Configuration,* both by Chris Grundemann, in the *Day One* library at http://www.juniper.net/dayone.

IS-IS Authentication

Most service providers use OSPF or IS-IS to allow fast internal convergence and scalability, as well as to use their traffic engineering capabilities with Multiprotocol Label Switching (MPLS). IS-IS is considered by some to be fundamentally more secure because it is not an IP-based routing protocol so it is more difficult to spoof and DoS than OSPF.

By default, the Junos implementation of IS-IS authentication inserts the MD5 checksum into link-state PDUs (LSPs), IIH PDUs, and complete and partial sequence number PDUs (CSNPs and PSNPs).

IS-IS also supports both simple and MD5 authentication in Level 1 and in Level 2. As we did in previous sections in this book, let's focus on MD5 because with simple authentication the key can easily be recovered by sniffing the protocol updates.

How to Enable IS-IS Authentication for Level 1

1. Start with a simple working ISIS configuration:

```
[edit protocols isis]
jweidley@mx80# show
interface ge-0/0/1.0 {
    level 2 disable;
}
interface lo0.0 {
    passive;
}
```

2. Set the authentication type to MD5:

```
[edit protocols isis]
jweidley@mx80# set level 1 authentication-type md5
```

3. Set the authentication key. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols isis]
jweidley@mx80# set level 1 authentication-key D5vw~\H,[bIOaG4
```

4. Verify the configuration:

```
[edit protocols isis]
jweidley@mx80# show
level 1 {
    authentication-key "$9$wHgoGjHmTFnHk9puOEhSrev7V"; ## SECRET-DATA
    authentication-type md5;
}
interface ge-0/0/1.0 {
    level 2 disable;
}
interface lo0.0 {
    passive;
}
```

How to Enable IS-IS Authentication for Hello Packet Exchanges

For increased security, a separate authentication key can also be configured to authenticate the IS-IS Hello protocol exchanges on a per interface basis.

1. Building on the example above, under the specific interface, set the authentication type:

```
[edit protocols isis]
jweidley@mx80# edit interface ge-0/0/1.0

[edit protocols isis interface ge-0/0/1.0]
jweidley@mx80# set level 1 hello-authentication-type md5
```

2. Set the authentication key. It is recommended that you use a key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols isis interface ge-0/0/1.0]
jweidley@mx80# set level 1 hello-authentication-key 4}QYWwR+^@V7^uf
```

3. Verify the entire configuration:

```
[edit protocols isis interface ge-0/0/1.0]
jweidley@mx80# up

[edit protocols isis]
jweidley@mx80# show
level 1 {
   authentication-key "$9$wHgoGjHmTFnHk9puOEhSrev7V"; ## SECRET-DATA
   authentication-type md5;
}
interface ge-0/0/1.0 {
   level 2 disable;
   level 1 {
      hello-authentication-key "$9$yDcKMXNds4JGdVjq.PzFn/CtIc"; ## SECRET-DATA
      hello-authentication-type md5;
   }
}
interface lo0.0 {
   passive;
}
```

NOTE Hitless Authentication Key Rollover for IS-IS is available in Junos 11.2 for M, MX, and T Series Devices. For more information see http://www.juniper.net/techpubs/en_US/junos14.1/topics/example/authentication-keychain-hitless-isis.html.

TIP The safest solution is to enable authentication for all IS-IS PDUs but it is important to be aware that there are configuration options to disable authentication for different PDU types. These options are `no-authentication-check`, `no-hello-authentication`, `no-psnp-authentication`, `no-csnp-authentication` and they were made available for interoperability with other vendors.

BGP Authentication

BGP is an exterior gateway protocol (EGP) that is used to connect Autonomous Systems (AS). Like the IGPs in the previous examples, BGP also supports MD5 authentication. Junos supports stronger algorithms than MD5 to accommodate higher security requirements, but for these examples let's use MD5.

MD5 is a widely used cryptographic hashing algorithm that generates a 128-bit hash value. The sending router inserts the configured MD5 hash in all transmitted BGP packets to its neighbors. When the receiving router receives the BGP packet it verifies this checksum before processing the packet contents.

One implementation consideration is whether to enable authentication globally, per group, or per peer. This flexibility can streamline your configuration and make authentication configuration and key changes more manageable in large environments.

How to Enable BGP Authentication Using MD5

TIP It is a recommended security practice to configure authentication for External BGP neighbors at the peer level so each key can be unique and not shared between service providers.

1. Start with a simple working BGP configuration:

```
[edit protocols bgp]
jweidley@MX960# show
group session-to-isp1 {
   type external;
   peer-as 65000;
   neighbor 192.168.11.1;
}
```

2. Enable authentication to the upstream facing peer router. It is recommended that you use an authentication key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols bgp]
jweidley@MX960# edit group session-to-isp1

[edit protocols bgp group session-to-isp1]
jweidley@MX960# set neighbor 192.168.11.1 authentication-key 4}QYWwR+^@V7^uf
```

3. Verify the entire configuration:

```
[edit protocols bgp group session-to-isp1]
jweidley@MX960# up

[edit protocols bgp]
jweidley@MX960# show
group session-to-isp1 {
   type external;
   peer-as 65000;
   neighbor 192.168.11.1 {
      authentication-key "$9$y8grWL4aUjkmcyoZjHTQEhSeM84aGUDHO1bsgJiH0BIhSe"; ## SECRET-DATA
   }
}
```

How to Enable BGP Authentication With Hitless Authentication Key Rollover

It's a good security practice to periodically change static passwords. This includes route authentication keys which can become more tedious or possibly unmanageable in large networks.

It's possible to change authentication keys without resetting any BGP peering sessions. This is referred to as *hitless authentication key rollover*.

Hitless authentication key rollover uses authentication *keychains*, which consist of multiple key IDs, shared secret passwords, and implementation dates and times. The keychains are then associated with the BGP neighbor session where the algorithm is configured.

1. Start with a simple working BGP configuration:

```
[edit protocols bgp]
jweidley@MX960# show
group session-to-core {
   type internal;
   local-address 10.10.10.170;
   neighbor 10.10.10.86;
}
```

2. Configure the keychain with the first key. Key IDs range from 0 thru 63 and set the time when the key should be active. It is recommended that you use an authentication key that is difficult to guess and that follows your organization's password complexity policy:

```
[edit protocols bgp]
jweidley@MX960# top edit security authentication-key-chains key-chain core-bgp-keychain

[edit security authentication-key-chains key-chain core-bgp-keychain]
jweidley@MX960# set key 0 secret D5vw~\H,[bI0aG4

[edit security authentication-key-chains key-chain core-bgp-keychain]
jweidley@m10# set key 0 start-time 2011-04-01.00:01
```

3. Configure the second key in the keychain. Repeat this step as many times as necessary:

```
[edit security authentication-key-chains key-chain core-bgp-keychain]
jweidley@MX960# set key 1 secret 4}QYWwR+^@V7^uf

[edit security authentication-key-chains key-chain core-bgp-keychain]
jweidley@MX960# set key 1 start-time 2011-07-01.00:01
```

4. Configure neighbor authentication to use the MD5 algorithm:

```
[edit security authentication-key-chains key-chain core-bgp-keychain]
jweidley@MX960# top edit protocols bgp group session-to-core

[edit protocols bgp group session-to-core]
jweidley@MX960# set neighbor 10.10.10.86 authentication-algorithm md5
```

5. Configure the neighbor to use the keychain:

```
[edit protocols bgp group session-to-core]
jweidley@MX960# set neighbor 10.10.10.86 authentication-key-chain core-bgp-keychain
```

6. And verify the configuration:

```
[edit protocols bgp group session-to-core]
jweidley@MX960# show
type internal;
local-address 10.10.10.170;
neighbor 10.10.10.86{
   authentication-algorithm md5;
   authentication-key-chain core-bgp-keychain;
}
```

```
[edit protocols bgp group session-to-core]
jweidley@MX960# top show security authentication-key-chains
key-chain core-bgp-keychain {
   description "automatic key management for BGP with core";
   key 0 {
      secret "$9$bp24ZDi.5z3iH/tpOREcylKxd"; ## SECRET-DATA
      start-time "2011-4-3.00:01:00 -0700";
   }
   key 1 {
      secret "$9$s9YJGQF//9tX7.PT3AtLxNVwg"; ## SECRET-DATA
      start-time "2011-7-1.00:01:00 -0700";
   }
}
```

NOTE Since the key rotation is based on date and time it is imperative to have a stable and reliable NTP time source configured. Also consider using the `key-chain tolerance` option which allows you to configure a clock skew.

MORE? For more details regarding hitless authentication key rollover visit https://www.juniper.net/documentation/en_US/junos14.1/topics/example/bgp-hitless-key-authentication.html.

See the section in this chapter on *Protecting the Routing Engine* and use the following sample firewall filter term to limit BGP.

Sample firewall filter term to limit BGP

```
set firewall family inet filter protect-re term allow-bgp from source-address <NEIGHBOR>
set firewall family inet filter protect-re term allow-bgp from destination-prefix-list <LOCAL IPs>
set firewall family inet filter protect-re term allow-bgp from protocol tcp
set firewall family inet filter protect-re term allow-bgp from destination-port bgp
set firewall family inet filter protect-re term allow-bgp then accept
```

Multicast Source Discovery Protocol (MSDP) Authentication

The Multicast Source Discovery Protocol (MSDP) is used to connect multicast routing domains. Each MSDP router establishes adjacencies with internal and external MSDP peers similar to the way BGP establishes peers. These peer routers inform each other about active sources within the domain.

By default, multicast routers accept and process any properly formatted MSDP messages from the configured peer address. This default behavior might violate the security policies in many organizations because MSDP messages can come from another routing domain beyond the control of the security practices of the multicast router's organization. Junos can authenticate MSDP messages using the Message Digest 5 (MD5) signature option for MSDP peering sessions. This authentication provides protection against spoofed packets being introduced into an MSDP peering session.

1. Start with a simple working MSDP configuration:

```
[edit protocols msdp]
jweidley@srx3400# show
local-address 10.10.10.1;
peer 10.10.10.2;
```

2. Use the `authentication-key` option at the peer level to set a strong shared secret key:

```
[edit protocols msdp]
jweidley@srx3400# set peer 10.10.10.2 authentication-key hPX!\/=sR14{

[edit protocols msdp]
jweidley@srx3400#


3. View the configuration:

[edit protocols msdp]
jweidley@srx3400# show
local-address 10.10.10.1;
peer 10.10.10.2 {
   authentication-key "$9$oqaGiHqPfzndV3/tOIRaZGimTFn9AuOF387"; ## SECRET-DATA
}

[edit protocols msdp]
jweidley@srx3400#
```

TIP Adding, removing, or changing an MSDP authentication key in a peering session resets the existing MSDP session and establishes a new session between the affected MSDP peers. This immediate session termination prevents excessive retransmissions and eventual session timeouts due to mismatched keys.

See the section in this chapter on *Protecting the Routing Engine* and use the following firewall filter term to limit MSDP.

Sample firewall filter term to limit MSDP

```
set firewall family inet filter protect-re term allow-msdp from source-address <NEIGHBOR>
set firewall family inet filter protect-re term allow-msdp from protocol tcp
set firewall family inet filter protect-re term allow-msdp from destination-port msdp
set firewall family inet filter protect-re term allow-msdp then accept
```

Multiprotocol Label Switching (MPLS)

There are many safeguards in a proper Multiprotocol Label Switching (MPLS) configuration that promotes logical segregation and increases security that are simply beyond the scope of this book. However, many MPLS-related vulnerabilities and attacks might be mitigated by simply not accepting labeled packets from, or sending labeled packets to, untrusted sources. So this section focuses on limiting MPLS conversations to only trusted sources.

Part of configuring MPLS is to enable `family mpls` under the interface. This configuration option tells the system not to drop MPLS protocol packets if they are received on the interface. MPLS should only be configured on the required interfaces to ensure that your device will only accept MPLS packets from expected routers.

MPLS must also be enabled under the Junos `protocols` stanza to notify the routing protocol daemon (rpd) which interfaces participate in which protocols. In some publicly available documentation you may see references to `interface all`. Although this command helps to get a working MPLS configuration, it could leave your devices vulnerable to some common MPLS attacks.

Figure 4.2 depicts the recommended configuration of MPLS by only enabling the necessary protocols on the required interfaces. Customer Edge (CE) facing interfaces do not directly participate in MPLS and should not be configured.

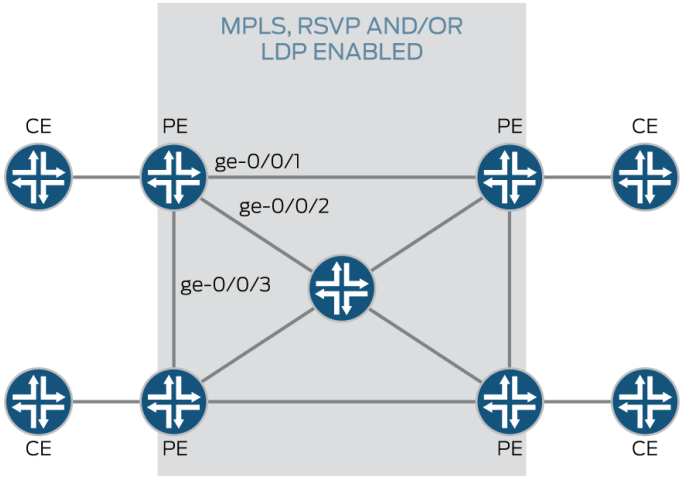


Figure 4.2 MPLS, RSVP, and/or LDP Enabled on Required Interfaces

How to Configure the MPLS Interfaces

As an example, let's configure MPLS on Figure 4.2's P and PE facing interfaces on PE1.

1. First you have to enable family MPLS at the interface level:

```
[edit interfaces]
jweidley@PE1# set ge-0/0/1 unit 0 family mpls

[edit interfaces]
jweidley@PE1# set ge-0/0/2 unit 0 family mpls

[edit interfaces]
jweidley@PE1# set ge-0/0/3 unit 0 family mpls
```

2. Then, under protocols MPLS, define only the P & PE facing interfaces:

```
[edit]
jweidley@PE1# edit protocols mpls

[edit protocols mpls]
jweidley@PE1# set interface ge-0/0/1.0

[edit protocols mpls]
jweidley@PE1# set interface ge-0/0/2.0

[edit protocols mpls]
jweidley@PE1# set interface ge-0/0/3.0

[edit protocols mpls]
jweidley@PE1# show
interface ge-0/0/1.0;
interface ge-0/0/2.0;
interface ge-0/0/3.0;
```

MORE? For detailed information on MPLS concepts and real-world configuration examples, see *This Week: Deploying MPLS* by Tim Fiola and Jamie Panagos, at http://www.juniper.net/dayone.

RSVP Authentication

RSVP protocol exchanges can also be authenticated to guarantee that only trusted neighbors participate in setting up reservations.

Junos supports MD5 for RSVP authentication. The computed message digest is transmitted to all neighbors, on a particular interface, to prevent forgery, message modification, and replay attacks. Be sure that the authentication key is difficult to guess and follows your organization's password complexity policy.

It is recommended that RSVP communication be configured only on specific interfaces that require authentication. Here's the whole configuration example:

```
[edit]
jweidley@PE1# edit protocols rsvp

[edit protocols rsvp]
jweidley@PE1# set interface ge-0/0/2.0 authentication-key 4{QYBxwR+$@V7!uf

[edit protocols rsvp]
jweidley@PE1# show
interface ge-0/0/2.0 {
   authentication-key "$9$d7V2aZGi.fzDi"; ## SECRET-DATA
}
```

NOTE It's important to mention that this feature does not provide confidentiality, meaning that the packet contents are not encrypted; it only authenticates trusted neighbors.

MORE? For additional information regarding RSVP Authentication see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

See the section in this chapter on *Protecting the Routing Engine* and use the following sample firewall filter term to limit RSVP.

Sample firewall filter term to limit RSVP

```
set firewall family inet filter protect-re term allow-rsvp from destination-prefix-list <LOCAL IPs>
set firewall family inet filter protect-re term allow-rsvp from protocol rsvp
set firewall family inet filter protect-re term allow-rsvp then accept
```

LDP Authentication

LDP authentication provides protection from spoofed TCP segments that could be introduced into connection streams for LDP sessions.

It is recommended that LDP communication be configured only on required interfaces and authenticated to protect against spoofing attacks. Be sure that the authentication key is difficult to guess and follows your organization's password complexity policy.

Here's the configuration in a nutshell:

```
[edit]
jweidley@PE1# edit protocols ldp session 10.10.10.85

[edit protocols ldp session 10.10.10.85]
jweidley@PE1# set authentication-key 4{QYBxwR+$@V7!uf

[edit protocols ldp session 10.10.10.85]
jweidley@PE1# show
authentication-key "$9$pJpguIcyrvL7Vev"; ## SECRET-DATA
```

NOTE LDP also supports stronger authentication algorithms as well as keychains for automatic key rotation as documented here: http://www.juniper.net/techpubs/en_US/junos14.1/topics/reference/configuration-statement/authentication-key-chains-edit-security.html. Also, check the Junos documentation for your specific platforms and code version.

MORE? For additional information regarding LDP Authentication see the *Junos Cookbook* (by Aviva Garrett, O'Reilly Media, 2006) at http://www.juniper.net/books.

See the section in this chapter on *Protecting the Routing Engine* and use the following sample firewall filter term to limit LDP.

Sample firewall filter term to limit LDP

```
set firewall family inet filter protect-re term allow-ldp from destination-address 224.0.0.2/32
set firewall family inet filter protect-re term allow-ldp from protocol udp
set firewall family inet filter protect-re term allow-ldp from destination-port ldp
set firewall family inet filter protect-re term allow-ldp then accept
```

Protecting the Routing Engine

The Routing Engine performs many different functions, from processing routing updates to running the Command Line Interface (CLI). As with any device, there are a finite amount of resources and these resources need to be protected to ensure stability and availability of the device.

Previous sections of this book showed you how to disable unsecure services, enable secure access, secure management services, apply the *least privilege* concept for logged in users, and enable authentication for routing protocols. The final step is protecting the routing engine by applying a firewall filter to permit only allowed source addresses to communicate with the services and protocols that are configured.

It's important to identify any traffic that hits the Routing Engine to determine what threat it could present to your device. In the last section, routing protocol authentication was discussed to ensure that routers exchange routes only with trusted peers. Although this helps protect routing protocols, it still does not completely prevent malicious or untrusted packets from reaching a particular process on the Routing Engine. For example, an attacker could still launch an attack against a router by targeting a particular protocol with forged packets. Although these packets will fail the authentication check, the attack may still consume router resources (CPU cycles and communication queues) on the Routing Engine, therefore making the attack successful to some extent. In order to avoid this, only protocol and control packets from trusted sources should be allowed through the firewall filter to reach the Routing Engine.

Creating a comprehensive firewall filter to protect your devices is a deep subject because there are site and device-specific considerations, and Junos has unique terminology and many options to provide functionality and granularity. That's why there's an entire *Day One* book dedicated to securing the Routing Engine (see the following reference). This section doesn't attempt to rehash the same concepts again, but instead is meant to provide a general overview of protecting the Routing Engine, define some necessary concepts, and provide implementation tips and tricks.

WARNING! The firewall filter that is designed in this section *does not cover* every management protocol, service, and routing protocol that was previously discussed in this book. The firewall filter that you implement on your own production devices will need to be customized to suit your environment. (That's why the relevant sections contain sample firewall filters so you can include or exclude them as required.)

CAUTION Although the firewall filter syntax is the same across all Junos platforms, due to differences in the underlying hardware (chipsets, ASICs, etc.) certain features are only supported on certain platforms. It is highly recommended that you review the firewall filter documentation for your specific platform prior to designing your filter.

MORE? For detailed background information and an excellent tutorial of using firewall filters to secure the routing engine, see *Day One: Securing the Routing Engine on M, MX, and T Series* by Douglas Hanks Jr., at http://www.juniper.net/dayone.

Designing Firewall Filters

There two basic methodologies for creating firewall filters:

- Default Permit: With this kind of filter you specify undesirable hosts, networks, or ports and protocols and deny them. The last term in the filter permits all other traffic.
- Default Deny: Firewall filters are created to permit traffic from trusted sources to the services and protocols that are configured on your device. The last term in the filter denies all other traffic.

The default deny method is the most secure and the most suitable for protecting the routing engine. This section uses the default deny method.

A default deny filter takes a little more thought and testing to ensure that all necessary services operate as expected. For example, the following list is a good place to start thinking about building your firewall filter and what to permit to the Routing Engine:

- Routing Protocols (BGP, OSPF, etc.)
- Access Services (SSH, J-Web, NETCONF, etc.)
- Management Services (SNMP, NTP, DNS, etc.)
- Diagnostic and Troubleshooting Protocols (ICMP, traceroute, etc.)

Other design tips to remember and consider are:

- Firewall filters are stateless. They do not have any knowledge of packets that were previously permitted or denied through the device.
- Firewall filters are processed in order from the top to the bottom. Although most Junos platforms process firewall filters in hardware at line rate, it's always a best practice to order your terms so that time sensitive protocols, like routing protocols, are positioned close to the top of the filter.
- It is recommended that you split each service out into its own term. This makes is easier to read, change, and troubleshoot.
- Deny terms can be placed at the top for known noisy protocols and services so your logs don't get cluttered with junk.

Firewall Filter Building Blocks

The firewall filter syntax is verbose and can sometimes be difficult to read and interpret. This section covers a few features that can make your firewall filters more readable and easily managed. Remember to download *Day One: Securing the Routing Engine* for a more detailed discussion: http://www.juniper.net/dayone.

Using Prefix-lists to Group Hosts or Networks

For readability purposes, most firewalls have a way to group like hosts and networks into a single object that can be referenced in your firewall rules. This allows you to reference a group of hosts/networks by a descriptive name.

Let's create a prefix list that contains the network management subnets. In Firewall filter terms this is done with a `prefix-list`:

1. Create a prefix-list called *mgmt-nets*:

```
[edit]
jweidley@MX240# edit policy-options prefix-list mgmt-nets
```

2. Add the IP subnets for the management subnets to the prefix-list:

```
[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# set 192.168.2.0/24

[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# set 192.168.4.0/24
```

3. Review the configuration:

```
[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# show
192.168.2.0/24;
192.168.4.0/24;
```

NOTE On flow-based platforms, like the J-Series and SRX devices, do not confuse prefix-lists with address-sets. Both are configurable but they are used for different purposes. Although the functionality is similar, address-sets are used in security policies while prefix-lists are used for packet-based filtering functions.

Using Apply-path to Build Dynamic Prefix-lists

A default deny firewall filter can be somewhat cumbersome to manage because you are permitting specific hosts and protocols and denying everything else. So what happens if you change a NTP server or add a new BGP peer? That's right, you have to remember to update the firewall filter or it's not going to work.

The `apply-path` feature in Junos can be used to dynamically create prefix-lists using matched patterns from specific sections of your Junos configuration. This reduces the number of occurrences of redundant information in your configuration, making it more readable and reducing the chances of configuration oversights.

For example, let's create a prefix-list called *ntp-servers* using the configuration from the *NTP* section of this book. Here's what our NTP configuration looks like:

```
[edit]
jweidley@MX240# show system ntp
boot-server 192.168.3.2;
authentication-key 1 type md5 value "$9$-kboZjHqKvMWLNs4"; ## SECRET-DATA
server 192.168.3.2 key 1 prefer; ## SECRET-DATA
```

```
server 192.168.33.2 key 1; ## SECRET-DATA
trusted-key 1;
source-address 172.25.44.132;
```

Now let's configure the prefix-list using `apply-path` so any changes to the NTP configuration will automatically update the prefix-list.

1. Create a prefix-list called *ntp-servers*:

```
[edit]
jweidley@MX240# edit policy-options prefix-list ntp-servers
```

2. Use the `apply-path` feature to match the NTP servers in our configuration:

```
[edit]
jweidley@MX240# set apply-path "system ntp server <*>"
```

3. Review the configuration:

```
[edit policy-options prefix-list ntp-servers]
jweidley@MX240# show
apply-path "system ntp server <*>";
```

Step 3 shows that the values for this prefix-list are being pulled from the NTP server statements configured under the `[system ntp]` hierarchy, but if you want to see the actual IP addresses you must use the `show | display inheritance` command, like this:

```
[edit policy-options prefix-list ntp-servers]
jweidley@MX240# show | display inheritance
##
## apply-path was expanded to:
##    192.168.3.2;
##    192.168.33.2;
##
apply-path "system ntp server <*>";
```

The greatest benefit of using `apply-path` is that Junos builds and maintains the dynamic prefix-list for you.

Creating Policers to Rate-limit Traffic

Policers are used to rate-limit traffic. They work in conjunction with firewall filters to set bandwidth limits on the traffic that is matched by the filter or term.

Configuring policers is pretty easy, but the trickiest part is determining the bandwidth limits.

NOTE Understanding how policers work, the algorithms used, and how to monitor them, is beyond the scope of this book. See the references after this brief tutorial for more details. This section simply focuses on how to use policers to protect the Routing Engine.

So, let's configure a policer with a 3MB limit.

1. Define the name of your policer. It's helpful to use a short descriptive name because it makes your firewall filter more readable. In this example let's call it *limit* and then insert the bandwidth maximum, 3m:

```
[edit firewall]
jweidley@MX240# edit policer limit-3m
```

2. Set the maximum bandwidth. The `bandwidth-limit` option is defined in bits per second, but luckily the Junos CLI allows you to specify more readable values: g (1,000,000,000), m (1,000,000), and k (1,000):

```
[edit firewall policer limit-3m]
jweidley@MX240# set if-exceeding bandwidth-limit 3m
```

3. The `burst-size-limit` option is mandatory because it specifies the maximum value that traffic can burst to before policing starts enforcing the average bandwidth. It is important to note that this value is specified in bytes:

```
[edit firewall policer limit-3m]
jweidley@MX240# set if-exceeding burst-size-limit 625k
```

4. Then define what you want to do with traffic that exceeds your configured thresholds, which is discarded, in our case:

```
[edit firewall policer limit-3m]
jweidley@MX240 # set then discard
```

5. View the configuration:

```
[edit firewall policer limit-3m]
jweidley@MX240# up

[edit firewall]
jweidley@MX240# show policer limit-3m
if-exceeding {
   bandwidth-limit 3m;
   burst-size-limit 625k;
}
then discard;
```

It's worth doing more research and practice on policing so you get a better understanding of how it works, as well as the impacts of the `burst-size-limit` option. If you set it too small, it could lead to over policing, and setting it too large could lead to under policing.

MORE? For more information on policers see *Day One: Securing the Routing Engine on M, MX, and T Series* by Douglas Hanks Jr. at http://www.juniper.net/dayone, or *Junos Enterprise Routing, 2nd Edition*, (by Southwick, Marschke & Reynolds, O'Reilly Media, 2008) at http://www.juniper.net/books.

Creating a Firewall Filter

Using the concepts described above, let's create a basic firewall filter to permit only the required services and traffic and then deny everything else to the Routing Engine.

1. First let's create the prefix-lists that are needed for BGP, OSPF, NTP, SNMP, RADIUS, local interfaces, and our management networks:

```
[edit]
jweidley@MX240# edit policy-options prefix-list bgp-neighbors

[edit policy-options prefix-list bgp-neighbors]
jweidley@MX240# set apply-path "protocols bgp group <*> neighbor <*>"

[edit policy-options prefix-list bgp-neighbors]
jweidley@MX240# up
```

```
[edit policy-options]
jweidley@MX240# edit prefix-list ipv4-interfaces

[edit policy-options prefix-list ipv4-interfaces]
jweidley@MX240# set apply-path "interfaces <*> unit <*> family inet address <*>"

[edit policy-options prefix-list ipv4-interfaces]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list ospf-all-routers

[edit policy-options prefix-list ospf-all-routers]
jweidley@MX240# set 224.0.0.5/32

[edit policy-options prefix-list ospf-all-routers]
jweidley@MX240# set 224.0.0.6/32

[edit policy-options prefix-list ospf-all-routers]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list ntp-servers

[edit policy-options prefix-list ntp-servers]
jweidley@MX240# set apply-path "system ntp server <*>"

[edit policy-options prefix-list ntp-servers]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list snmp-servers

[edit policy-options prefix-list snmp-servers]
jweidley@MX240# set apply-path "snmp community <*> clients <*>"

[edit policy-options prefix-list snmp-servers]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list mgmt-nets

[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# set 192.168.3.0/24

[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# set 192.168.33.0/24

[edit policy-options prefix-list mgmt-nets]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list radius-servers

[edit policy-options prefix-list radius-servers]
jweidley@MX240# set apply-path "system radius-server <*>"

[edit policy-options prefix-list radius-servers]
jweidley@MX240# up

[edit policy-options]
jweidley@MX240# edit prefix-list localhost
```

```
[edit policy-options prefix-list localhost]
jweidley@MX240# set 127.0.0.1/32

[edit policy-options prefix-list localhost]
jweidley@MX240# top
```

2. Create terms to permit the dynamic routing protocols you have configured. This example configures terms for BGP and OSPF:

```
[edit]
jweidley@MX240# edit firewall family inet filter protect-re

[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-bgp

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# set from source-prefix-list bgp-neighbors

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# set from destination-prefix-list ipv4-interfaces

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# set from protocol tcp

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# set from destination-port bgp

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-bgp]
jweidley@MX240# up

[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-ospf

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# set from source-prefix-list ipv4-interfaces

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# set from destination-prefix-list ospf-all-routers

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# set from destination-prefix-list ipv4-interfaces

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# set from protocol ospf

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-ospf]
jweidley@MX240# up
```

3. Next let's create the term to support secure remote access via Secure Shell Protocol (SSH) from the trusted management networks:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-ssh

[edit firewall family inet filter protect-re term allow-ssh]
jweidley@MX240# set from source-prefix-list mgmt-nets
```

```
[edit firewall family inet filter protect-re term allow-ssh]
jweidley@MX240# set from protocol tcp

[edit firewall family inet filter protect-re term allow-ssh]
jweidley@MX240# set from destination-port ssh

[edit firewall family inet filter protect-re term allow-ssh]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-ssh]
jweidley@MX240# up
```

4. Permit SNMP access from authorized servers for network monitoring:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-snmp

[edit firewall family inet filter protect-re term allow-snmp]
jweidley@MX240# set from source-prefix-list snmp-servers

[edit firewall family inet filter protect-re term allow-snmp]
jweidley@MX240# set from protocol udp

[edit firewall family inet filter protect-re term allow-snmp]
jweidley@MX240# set from destination-port snmp

[edit firewall family inet filter protect-re term allow-snmp]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-snmp]
jweidley@MX240# up
```

5. Permit NTP synchronization with configured servers:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-ntp

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# set from source-prefix-list ntp-servers

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# set from source-prefix-list localhost

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# set from protocol udp

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# set from destination-port ntp

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-ntp]
jweidley@MX240# up
```

6. Allow RADIUS responses only from the configured RADIUS servers. Notice that the filter is specific to the source port, which matches the responses since this is an input filter:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-radius

[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# set from source-prefix-list radius-servers
```

```
[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# set from protocol udp

[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# set from source-port radius

[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# set from source-port radacct

[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-radius]
jweidley@MX240# up
```

7. For troubleshooting purposes, let's allow specific useful ICMP types and UDP ports for traceroute, but first block all fragmented ICMP packets:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term icmp-frags

[edit firewall family inet filter protect-re term icmp-frags]
jweidley@MX240# set from protocol icmp

[edit firewall family inet filter protect-re term icmp-frags]
jweidley@MX240# set from is-fragment

[edit firewall family inet filter protect-re term icmp-frags]
jweidley@MX240# set then syslog discard

[edit firewall family inet filter protect-re term icmp-frags]
jweidley@MX240# up

[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-icmp

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set from protocol icmp

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set from icmp-type echo-request

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set from icmp-type echo-reply

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set from icmp-type unreachable

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set from icmp-type time-exceeded

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-icmp]
jweidley@MX240# up

[edit firewall family inet filter protect-re]
jweidley@MX240# edit term allow-traceroute

[edit firewall family inet filter protect-re term allow-traceroute]
jweidley@MX240# set from protocol udp
```

```
[edit firewall family inet filter protect-re term allow-traceroute]
jweidley@MX240# set from destination-port 33434-33523

[edit firewall family inet filter protect-re term allow-traceroute]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term allow-traceroute]
jweidley@MX240# up
```

8. Create a term that will permit all established sessions. The `tcp-established` keyword matches TCP packets with either the ACK or RST bit set:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term tcp-established

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# set from protocol tcp

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# set from source-port ssh

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# set from source-port bgp

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# set from tcp-established

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# set then accept

[edit firewall family inet filter protect-re term tcp-established]
jweidley@MX240# up
```

9. Finally, configure the default deny term to discard and log all traffic. The `log` option saves the packet header information in a buffer on the Packet Forwarding Engine (PFE) and the `syslog` option stores the packet header information on the Routing Engine:

```
[edit firewall family inet filter protect-re]
jweidley@MX240# edit term default-deny

[edit firewall family inet filter protect-re term default-deny]
jweidley@MX240# set then syslog log discard

[edit firewall family inet filter protect-re term default-deny]
jweidley@MX240# top
```

10. And let's review the firewall configuration:

```
[edit]
jweidley@MX240# show policy-options
prefix-list bgp-neighbors {
    apply-path "protocols bgp group <*> neighbor <*>";
}
prefix-list ipv4-interfaces {
    apply-path "interfaces <*> unit <*> family inet address <*>";
}
prefix-list ospf-all-routers {
    224.0.0.5/32;
    224.0.0.6/32;
}
prefix-list ntp-servers {
```

```
    apply-path "system ntp server <*>";
}
prefix-list snmp-servers {
    apply-path "snmp community <*> clients <*>";
}
prefix-list mgmt-nets {
    192.168.3.0/24;
    192.168.33.0/24;
}
prefix-list radius-servers {
    apply-path "system radius-server <*>";
}
prefix-list localhost {
    127.0.0.1/32;
}

[edit]
jweidley@MX240# show firewall | no-more
family inet {
    filter protect-re {
        term allow-bgp {
            from {
                source-prefix-list {
                    bgp-neighbors;
                }
                destination-prefix-list {
                    ipv4-interfaces;
                }
                protocol tcp;
                destination-port bgp;
            }
            then accept;
        }
        term allow-ospf {
            from {
                source-prefix-list {
                    ipv4-interfaces;
                }
                destination-prefix-list {
                    ospf-all-routers;
                    ipv4-interfaces;
                }
                protocol ospf;
            }
            then accept;
        }
        term allow-ssh {
            from {
                source-prefix-list {
                    mgmt-nets;
                }
                protocol tcp;
                destination-port ssh;
            }
            then accept;
        }
        term allow-snmp {
            from {
                source-prefix-list {
                    snmp-servers;
                }
                protocol udp;
                destination-port snmp;
```

```
            }
            then accept;
        }
        term allow-ntp {
            from {
                source-prefix-list {
                    ntp-servers;
                    localhost;
                }
                protocol udp;
                destination-port ntp;
            }
            then accept;
        }
        term allow-radius {
            from {
                source-prefix-list {
                    radius-servers;
                }
                protocol udp;
                source-port [ radius radacct ];
            }
            then accept;
        }
        term icmp-frags {
            from {
                is-fragment;
                protocol icmp;
            }
            then {
                syslog;
                discard;
            }
        }
        term allow-icmp {
            from {
                protocol icmp;
                icmp-type [ echo-request echo-reply unreachable time-exceeded ];
            }
            then accept;
        }
        term allow-traceroute {
            from {
                protocol udp;
                destination-port 33434-33523;
            }
            then accept;
        }
        term tcp-established {
            from {
                protocol tcp;
                source-port [ ssh bgp ];
                tcp-established;
            }
            then accept;
        }
        term default-deny {
            then {
                log;
                syslog;
                discard;
            }
        }
    }
}
```

Creating a Firewall Filter with Rate-Limiting

Rate-limiting provides another layer of protection in a filter. In addition to allowing only specific protocols from specific hosts, you can also rate limit the amount of allowed traffic to reasonable levels, ensuring that authorized hosts cannot flood the Routing Engine.

To avoid duplication, let's build on the firewall filter from the previous section by adding policers to the existing firewall terms to rate-limit specific types of traffic. A new term will be added, as well, to protect the Routing Engine from a TCP SYN Flood attack.

1. First let's configure a couple policers that will limit traffic at different rates:

```
[edit firewall]
jweidley@MX240#edit policer limit-10m

[edit firewall policer limit-10m]
jweidley@MX240#set if-exceeding bandwidth-limit 10m

[edit firewall policer limit-10m]
jweidley@MX240#set if-exceeding burst-size-limit 625k

[edit firewall policer limit-10m]
jweidley@MX240#set then discard

[edit firewall policer limit-10m]
jweidley@MX240#up

[edit firewall]
jweidley@MX240#edit policer limit-3m

[edit firewall policer limit-3m]
jweidley@MX240#set if-exceeding bandwidth-limit 3m

[edit firewall policer limit-3m]
jweidley@MX240#set if-exceeding burst-size-limit 15k

[edit firewall policer limit-3m]
jweidley@MX240#set then discard

[edit firewall policer limit-3m]
jweidley@MX240#up


[edit firewall]
jweidley@MX240#edit policer limit-1m

[edit firewall policer limit-1m]
jweidley@MX240#set if-exceeding bandwidth-limit 1m

[edit firewall policer limit-1m]
jweidley@MX240#set if-exceeding burst-size-limit 15k

[edit firewall policer limit-1m]
jweidley@MX240#set then discard

[edit firewall policer limit-1m]
jweidley@MX240#up

[edit firewall]
jweidley@MX240#edit policer limit-100k
```

```
[edit firewall policer limit-100k]
jweidley@MX240#set if-exceeding bandwidth-limit 100k

[edit firewall policer limit-100k]
jweidley@MX240#set if-exceeding burst-size-limit 15k

[edit firewall policer limit-100k]
jweidley@MX240#set then discard

[edit firewall policer limit-100k]
jweidley@MX240#up

[edit firewall]
jweidley@MX240#edit policer limit-32k

[edit firewall policer limit-32k]
jweidley@MX240#set if-exceeding bandwidth-limit 32k

[edit firewall policer limit-32k]
jweidley@MX240#set if-exceeding burst-size-limit 15k

[edit firewall policer limit-32k]
jweidley@MX240#set then discard

[edit firewall policer limit-32k]
jweidley@MX240#top
```

2. To further protect the Routing Engine let's add policers to our management protocols to ensure they behave appropriately. SSH traffic will be limited to 10Mbps. This may seem high for some environments, but this value allows for transferring large files, like new Junos images, to the Routing Engine, via Secure Copy Protocol (SCP):

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-ssh then policer limit-10m
```

3. Let's limit SNMP traffic to 1Mbps. This value should allow network management to safely pull many OIDs without impacting the Routing Engine:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-snmp then policer limit-1m
```

4. NTP traffic only involves a few packets every hour, so it will be limited to 32Kbps:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-ntp then policer limit-32k
```

5. And RADIUS traffic shouldn't take much bandwidth, so in this example traffic will be limited to 32Kbps:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-radius then policer limit-32k
```

How to Protect Against Flood Attacks

Flood attacks try to overwhelm the target device in an attempt to consume all system resources. Common flood attacks have been successful using ICMP and TCP SYN packets, but flood attacks using other packet types are definitely possible. This section applies policers to protocols that are commonly used in flooding attacks.

1. The first policer to add is for the `tcp-established` term. The `tcp-established` keyword in the firewall filter essentially matches any packet that has the ACK or RST bits set. This could make the Routing Engine susceptible to a flood attack. So, let's apply a 10Mbps policer to that term:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term tcp-established then policer limit-10m
```

2. Troubleshooting protocols should be allowed to assist in fault isolation, but bandwidth restrictions should be applied so they do not consume too many system resources. ICMP traffic will be limited to 1Mbps, which should be suitable in most cases for both troubleshooting and performance monitoring:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-icmp then policer limit-1m
```

3. Traceroute traffic will be limited to 1Mbps:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#set term allow-traceroute then policer limit-1m
```

4. Although SYN attacks are probably the most well known TCP flood attacks, other TCP control flags can also be used. Let's apply a policer to TCP control packets that have the RST, FIN, and SYN (as long as there isn't also an ACK) flags set:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#edit term synflood-protect

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set from source-prefix-list bgp-neighbors

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set from source-prefix-list mgmt-nets

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set from protocol tcp

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set from tcp-flags "(syn & !ack) | fin | rst"

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set then policer limit-100k

[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#set then accept
```

5. By default new terms are added to the end of the filter and to make SYN Flood protection effective, the term has to be placed at the top of the filter:

```
[edit firewall family inet filter protect-re term synflood-protect]
jweidley@MX240#up

[edit firewall family inet filter protect-re]
jweidley@MX240#insert term synflood-protect before term allow-bgp
```

6. Let's review the firewall filter and policer configuration:

```
[edit firewall family inet filter protect-re]
jweidley@MX240#up 2

[edit firewall]
```

```
jweidley@MX240#show | no-more
family inet {
   filter protect-re {
      term synflood-protect {
         from {
            source-prefix-list {
               bgp-neighbors;
               mgmt-nets;
            }
            protocol tcp;
            tcp-flags "(syn & !ack) | fin | rst";
         }
         then {
            policer limit-100k;
            accept;
         }
      }
      term allow-bgp {
         from {
            source-prefix-list {
               bgp-neighbors;
            }
            destination-prefix-list {
               ipv4-interfaces;
            }
            protocol tcp;
            destination-port bgp;
         }
         then accept;
      }
      term allow-ospf {
         from {
            source-prefix-list {
               ipv4-interfaces;
            }
            destination-prefix-list {
               ospf-all-routers;
               ipv4-interfaces;
            }
            protocol ospf;
         }
         then accept;
      }
      term allow-ssh {
         from {
            source-prefix-list {
               mgmt-nets;
            }
            protocol tcp;
            destination-port ssh;
         }
         then {
            policer limit-10m;
            accept;
         }
      }
      term allow-snmp {
         from {
            source-prefix-list {
               snmp-servers;
            }
            protocol udp;
            destination-port snmp;
```

```
    }
    then {
        policer limit-1m;
        accept;
    }
}
term allow-ntp {
    from {
        source-prefix-list {
            ntp-servers;
            localhost;
        }
        protocol udp;
        destination-port ntp;
    }
    then {
        policer limit-32k;
        accept;
    }
}
term allow-radius {
    from {
        source-prefix-list {
            radius-servers;
        }
        protocol udp;
        source-port [ radius radacct ];
    }
    then {
        policer limit-32k;
        accept;
    }
}
term icmp-frags {
    from {
        is-fragment;
        protocol icmp;
    }
    then {
        syslog;
        discard;
    }
}
term allow-icmp {
    from {
        protocol icmp;
        icmp-type [ echo-request echo-reply unreachable time-exceeded ];
    }
    then {
        policer limit-1m;
        accept;
    }
}
term allow-traceroute {
    from {
        protocol udp;
        destination-port 33434-33523;
    }
    then {
        policer limit-1m;
        accept;
    }
}
```

```
        term tcp-established {
            from {
                protocol tcp;
                source-port [ ssh bgp ];
                tcp-established;
            }
            then {
                policer limit-10m;
                accept;
            }
        }
        term default-deny {
            then {
                log;
                syslog;
                discard;
            }
        }
    }
}
policer limit-10m{
    if-exceeding{
        bandwidth-limit 10m;
        burst-size-limit 625k;
    }
    then discard;
}
policer limit-1m {
    if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
    }
    then discard;
}
policer limit-100k{
    if-exceeding{
        bandwidth-limit 100k;
        burst-size-limit 15k;
    }
    then discard;
}
policer limit-32k{
    if-exceeding{
        bandwidth-limit 32k;
        burst-size-limit 15k;
    }
    then discard;
}
```

Implementing Firewall Filters

Using firewall filters is a two-step process: first you create your filters and then you need to apply them to an interface to filter traffic in a specific direction.

Since the objective here is to secure the Routing Engine, let's apply our firewall filters to the loopback interface (lo0). The loopback interface is a logical interface that is associated with the Routing Engine, so in order to protect the Routing Engine the firewall filter needs to be applied only to the loopback interface and not to all of the transit interfaces, as illustrated in Figure 4.3.

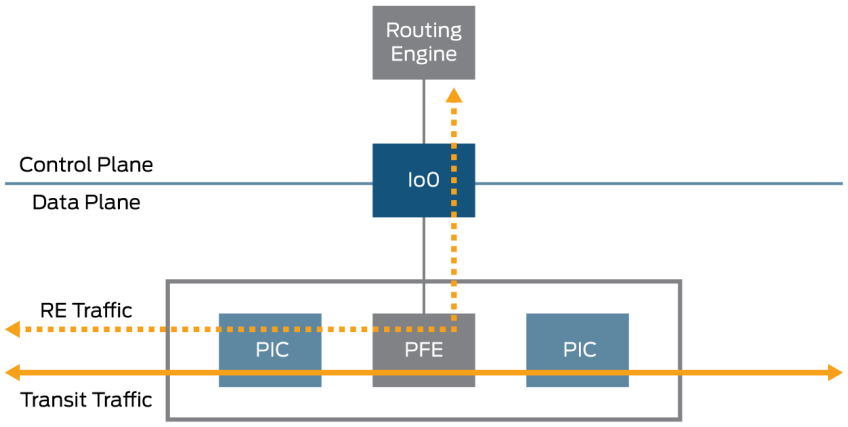


Figure 4.3 Transit and RE Traffic

Firewall filters must be told in which direction to inspect traffic, and there are two directions in which to apply the filters:

- Input: Packets are matched against the firewall filter as they enter the interface from the network.
- Output: Packets are matched against the firewall filter as they leave the interface prior to reaching the network.

To protect the Routing Engine let's implement our firewall filters to inspect traffic as it comes into the lo0 interface (input) as shown by Figure 4.4.

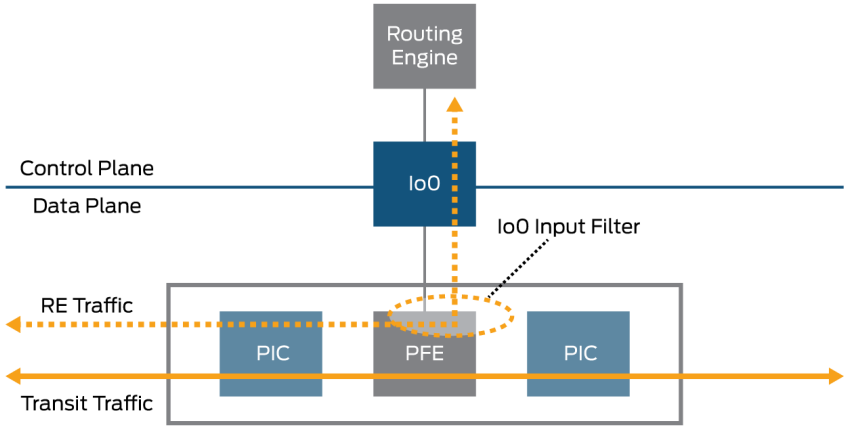


Figure 4.4 Transit Traffic and RE Traffic With lo0 Input Filter

In our example here the *protect-re* filter is applied to the lo0.0 interface:

```
[edit interfaces lo0]
jweidley@MX240# show
unit 0 {
    family inet {
        filter {
            input protect-re;
        }
        address 10.10.10.170/32;
    }
}
```

Tips and Self Preservation Techniques

It doesn't matter how much thought and testing you put into your firewall filter, when it comes time to actually commit the changes it's natural to be a little nervous so be conservative and cautious. Luckily Junos has some built-in features to help ease the anxiety, and there are a couple of tricks that can increase your chances of success.

The first and most obvious implementation tip is committing your firewall filters from the console port. This reduces the chances of locking out access to the device.

Since console access isn't always available, the second tip is that you can use the `commit confirmed` command, which provides a two-step commit process. After the command is entered, your changes are implemented but will not remain permanent until you issue a second `commit` within the specified time. If the second `commit` isn't entered before the specified time the configuration is automatically rolled back. This is a great command to prevent locking yourself out of your device.

The default confirmed rollback time is 10 minutes, which can seem like an eternity if you've accidently locked yourself out of the device and are blocking routing updates. Luckily, the confirmed timeout is configurable. Two minutes is a reasonable rollback time because it gives you some time to verify remote access, routing adjacencies, etc. Let's look at what an automatic rollback would look like:

```
[edit interfaces lo0 unit 0]
jweidley@MX240# commit confirmed 2
commit confirmed will be automatically rolled back in 2 minutes unless confirmed
commit complete

# commit confirmed will be rolled back in 2 minutes
[edit interfaces lo0 unit 0]
jweidley@MX240#

Broadcast Message from root@MX240
       (no tty) at 1:47 EDT...

Commit was not confirmed; automatic rollback complete.

[edit interfaces lo0 unit 0]
jweidley@MX240#
```

The third implementation tip is to insert a 'temporary' `default-permit` term with logging enabled above your `default-deny` term. This way you can monitor the default permit log to see if you forgot to include something in your firewall filter. Let's do it like this:

```
[edit firewall family inet filter protect-RE]
lab@MX240# show
. . . .
term temp-default-permit {
   then {
      log;
      accept;
   }
}
term default-deny {
   then {
      log;
      syslog;
      reject;
   }
}
```

You can then use the `show firewall log` command to see if important traffic was missed. Chances are the firewall log will not be completely "clean," so review the output closely:

```
jweidley@MX240> show firewall log
Log :
Time     Filter   Action Interface  Protocol Src Addr     Dest Addr
20:09:26  protect-re A    fxp0.0    IGMP    172.25.46.190 224.0.0.1
20:09:26  protect-re A    fxp0.0    UDP     172.25.46.6   172.25.46.255
20:09:19  protect-re A    fxp0.0    IGMP    172.25.45.140 224.0.0.1
20:09:17  protect-re A    fxp0.0    IGMP    172.25.46.96  224.0.0.1
20:09:11  protect-re A    fxp0.0    IGMP    172.25.46.180 224.0.0.1
20:09:07  protect-re A    fxp0.0    IGMP    172.25.46.182 224.0.0.1
20:09:05  protect-re A    fxp0.0    UDP     172.25.46.15  172.25.46.255
```

You can also use the `show firewall log detail` command to show the source and destination ports. Once you're satisfied that important traffic is not being denied, the temporary default permit term can be removed.

The last implementation tip is to use counters as a method to see how much of the specific traffic the device is receiving. The internal log buffers can only store so much data and sometimes when you run the `show firewall log` command you get the perception that your device is being "flooded" with traffic. Using a combination of counters and the `clear` and `show firewall counters` commands you can get a better representation of how much data is being denied.

```
jweidley@MX240> show firewall

Filter: protect-re
Counters:
Name                Bytes           Packets
count-syn           4840069          55160
count-ping          69624648         229862
default-deny        4805044704       28671684
```

Preserving System Hardening

Now that you have spent a significant amount of time hardening your Junos devices and improving the security posture of your network, you should consider taking steps to ensure that authorized engineers can't make configuration changes that would compromise the device. This section discusses a few options to assist you in maintaining a consistent security posture.

'protect'

Junos 11.2 enables you to protect the device configuration from being easily modified or deleted by other users. This can be accomplished by using the `protect` command in the configuration mode of the CLI. Likewise, you can also unprotect a protected configuration by using the `unprotect` command.

The `protect` command can be used at any level of the configuration hierarchy and will prevent users from deleting, modifying, inserting, renaming, copying, activating, deactivating, or annotating a protected configuration hierarchy.

Let's take a look at the `protect` command in action. In this example the firewall filter on lo0 will be protected and then we'll try to deactivate it so we can see the error message that is displayed:

```
[edit]
jweidley@srx210# show interfaces lo0
description “loopback interface”;
unit 0 {
    family inet {
        filter {
            input protect-re;
        }
        address 192.168.1.1/24;
    }
}

[edit]
jweidley@srx210# protect interfaces lo0 unit 0 family inet filter

[edit]
jweidley@srx210# show interfaces lo0
description “loopback interface”;
unit 0 {
    family inet {
        protect: filter {
            input protect-re;
        }
        address 192.168.1.1/24;
    }
}

[edit]
jweidley@srx210# commit comment “protected lo0 filter”
commit complete

[edit]
jweidley@srx210# deactivate interfaces lo0 unit 0 family inet filter
warning: [interfaces lo0 unit 0 family inet filter] is protected, ‘interfaces lo0 unit 0 family inet
filter’ cannot be deactivated

[edit]
jweidley@srx210#
```

Using `protect` isn't 100% foolproof, an engineer with the appropriate privileges could still `unprotect` that section of the configuration but it would require knowledge of the `unprotect` command and multiple commits to the configuration. `Protect` works well in preventing unintentional changes and serves as a possible deterrent for unauthorized changes.

MORE? See *Protecting the Junos OS Configuration from Modification or Deletion* for more details and examples at http://www.juniper.net/techpubs/en_US/junos14.2/topics/task/configuration/junos-cli-configuration-protecting.html.

Configuration Automation: Commit Scripts

Commit scripts are a component of Junos automation that allow you to customize the configuration validation process in accordance with your network standards and best common practices. Junos integrates commit script seamlessly into its commit process, allowing the scripts to check or modify the configuration prior to the final validation performed by Junos.

Junos Automation is a deep topic and is the subject of four *Day One/This Week* books, and a forthcoming book from O'Reilly Media: *Automating Junos Administration* (April 2016). The intention here is to provide a very simple functional example of how to use a commit script to solve a specific problem.

For this example, a company's best common practice is to not use unencrypted management services (such as Telnet & FTP). To show two different user feedback options, a warning message will be displayed when Telnet is enabled, and when FTP is enabled an error message will be displayed and the commit process will fail.

Here is the commit script:

```
jweidley@ex3200> file show /var/db/scripts/commit/security-baseline.slax
version 1.0;
ns junos = "http://xml.juniper.net/junos/*/junos";
ns xnm = "http://xml.juniper.net/xnm/1.1/xnm";
ns jcs = "http://xml.juniper.net/junos/commit-scripts/1.0";
import "../import/junos.xsl";


match configuration {
       /* Telnet Check */
       var $telnet-enabled = system/services/telnet;

       if ( $telnet-enabled ) {
              <xnm:warning> {
                     <message> "Telnet is an insecure service and should NOT be enabled!";
              }
       }

       /* FTP Check */
       var $ftp-enabled = system/services/ftp;

       if ( $ftp-enabled ) {
              <xnm:error> {
                     <message> "FTP is an insecure service and should NOT be enabled!";
              }
       }

}
/* End of script */
```

Let's test the script by enabling and Telnet and FTP and ensure that it provides the desired results:

```
jweidley@ex3200> show configuration system scripts
commit {
   file security-baseline.slax;
}

jweidley@ex3200> configure private
warning: uncommitted changes will be discarded on exit
Entering configuration mode

[edit]
jweidley@ex3200# set system services telnet

[edit]
jweidley@ex3200# commit
warning: Telnet is an insecure service and should NOT be enabled!

[edit]
 jweidley@ex3200#
```

So far the commit script is working exactly as we expected. After enabling Telnet the commit is successful and our custom warning message is displayed. Now let's enable FTP:

```
[edit]
jweidley@ex3200# set system services ftp

[edit]
jweidley@ex3200# commit
warning: Telnet is an insecure service and should NOT be enabled!
error: FTP is an insecure service and should NOT be enabled!
error: 1 error reported by commit scripts
error: commit script failure

[edit]
jweidley@ex3200#
```

During the second commit the Telnet warning is displayed again. That is expected behavior because commit scripts are executed every time a commit is done. Then our custom error message regarding FTP is displayed and then system-generated messages are received about the commit failure.

Commit scripts are an incredibly powerful tool that can help solve many challenges. Please take the time to explore the functionality and flexibility they provide.

TIP To reduce the chances of commit scripts being disabled, ensure you restrict modification of the *system scripts commit* configuration hierarchy to trusted engineers.

MORE? For books in the *Day One Automation* series see http://www.juniper.net/us/en/training/jnbooks/day-one/automation-series/. For automation examples, see: https://techwiki.juniper.net/Automation_Scripting/030_Examples.

Junos Space: Device Templates

Junos Space Network Management Platform provides comprehensive management of Juniper devices with broad fault, configuration, accounting, performance, and security management capability, same day support for new devices and Junos OS releases, a task-specific user interface, and northbound APIs for integration with existing network management systems (NMS). There are many useful features in Junos Space but in this section we are only going to discuss device templates and system of record as they pertain to system hardening.

Device templates are an excellent way to create base configurations for new devices and to standardize common configurations on all deployed devices in your enterprise. Multiple device templates can be assigned to a single device that allows you to have a management template, hardening template, routing template, etc.

The network you are administering is the system of record (SOR) —each device defines its own official state— you may prefer to have the Junos Space Network Management database contain the official state of the network, enabling you to restore that official state if unwanted changes are made to a device. You can designate the Junos Space Network Management database values as having precedence over any values configured locally at a device. In this scenario, Junos Space is the system of record. It contains the configurations that the Junos Space administrator considers best for the network devices. If a commit operation is executed on a network device,

Junos Space receives a notification and the administrator can choose whether or not to overwrite the device's local changes by reapplying device template settings to the device.

MORE? Junos Space provides comprehensive device management of Junos devices and this section just scratches the surface of two features. If this functionality sounds like it could be beneficial to your network, review the Space documentation for more details regarding functionality, implementation, and caveats. For more information regarding Junos Space, please see Juniper's documentation: http://www.juniper.net/techpubs/en_US/junos-space14.1/platform/information-products/pathway-pages/user-guide.html

Summary

This book has covered many different aspects of securing your Junos devices, and hopefully you can appreciate the inherent security features built in to Junos – the flexibility, the power of a common operating system, and most of all, the ability to secure your Junos devices while still maintaining required functionality.

When developing a hardening template use any of these techniques to meet your network's unique security requirements. Consult your security policy for guidance and direction. Research how the changes could impact other teams, normal operations, and emergency scenarios. Test changes in a lab and educate all engineers of the changes in configuration and behavior. Most of all, be methodical, be cautious, and continuously monitor.

Congratulations on increasing the security posture of your device!

Appendices

Appendix A: Juniper's U.S. Government Certifications

Juniper Networks expends great effort and resources to ensure the appropriate devices get the required U.S. Government certifications. The following outlines the process, but first, for certification-related or technical information, contact the Director of Federal Strategic Initiatives at uc-apl@juniper.net.

The Path to the Approved Products List

The U.S. Government requires several certifications before listing networking devices on the appropriate Approved Products List (APL), a prerequisite for sales to government entities. In an effort to reduce the number of APLs within the Department of Defense (DoD), the DoD has mandated a single APL – the Unified Capabilities Approved Products List (UC-APL). Each service may, if required, add service-specific requirements above those found in the Unified Capabilities Requirements document (UCR). These additional requirements are provided by the service's testing laboratory and are included in the negotiations between the government and Juniper Networks.

The UCR details a great number of requirements but categorizes them for specific technologies (general Network Appliance – NA; Router – R; or Layer 3 switch – LS). Therefore, some requirements designated for routers may not be applicable to general network appliances, but if they are, then the requirement will be marked for both devices. The UC Test Plan, written by the Joint Interoperability Test Command (JITC), takes the requirements and combines them with detailed procedures to ensure repeatability regardless of what equipment is tested. All of the UC documents can be accessed from the Uniform Capabilities Certification Office (UCCO) home page (then follow to Policies and Procedures, and then to Key Documents and Requirements).

The UCR list requires that devices be Federal Information Processing Standards (FIPS) and National Information Assurance Partnership (NIAP) certified prior to getting tested. FIPS certification, conducted by National Institute of Standards and Technology (NIST) certified laboratories, validates encryption algorithms while NIAP uses Protection Profiles to ensure the device meets specific Information Assurance (IA) standards. Completed FIPS 140-2 certification is required, although pending NIAP certification is sufficient to begin UC testing. NIST generates certificates and Security Policy documents and NIAP generates Security Target and Validation Report documents, and Common Criteria Certificates.

UC testing is conducted at various government laboratories, including the Defense Information Systems Agency (DISA) Joint Interoperability Test Command (JITC) in Ft. Huachuca, AZ, the Army Technology Integration Center (TIC) in Ft. Huachuca, AZ, JITC in Indian Head, MD, and other labs across the country. The Juniper Federal Certification Effort is supported by Resident Test Engineers, part of the Customer Service organization. Juniper products are scheduled by the Federal Strategic Initiatives office after negotiating a government sponsor, an appropriate test facility, the required number and types of devices (including firmware version), shipping schedules, travel requirements, and Demo Pool support. The test engineers support the government's evaluation at whatever laboratory was selected.

Once the evaluation is complete, any shortcomings are addressed in Test Deficiency Reports (TDRs). TDRs are submitted to (DISA) for adjudication. If the TDR is not closed, Juniper is required to file a Plan of Action and Milestones (POAM) that details when the device under test (DUT) will meet the requirement and any mitigating actions required prior to full compliance; in addition to any others, mitigating actions become conditions of fielding that must be implemented to field the DUT in an approved manner.

A successful evaluation generates a Certification Letter, a DoD Information Assurance Certification and Accreditation Process (DIACAP) Scorecard, and inclusion on the UC-APL. The DIACAP scorecard is available to government and military personnel who must send a Common Access Card (CAC)-signed email to ucco@disa.mil. The DIACAP scorecard is required for network accreditation when the DUT is fielded. By applying the security configuration used during testing, the fielding organization can use the completed DIACAP scorecard to shorten the accreditation process.

To aid configuration, and ultimately network accreditation, Juniper provides a *Secure Deployment Guide* for each tested device. The guides provide a single-source document that describes the tested configuration and provides examples, the tested configuration, and any scripts used. The guide is offered in PDF format to reduce file size and ensure cross-platform compatibility. Contact the Director of Federal Strategic Initiatives for more information.

The following websites may aid further exploration:

- UCCO Home Page:
  http://www.disa.mil/ucco/index.html
- DISA APL Process Guide:
  http://www.disa.mil/ucco/apl_process.html?panel=1#A_Services
- UC APL Testing Centers of Excellence:
  http://www.disa.mil/ucco/testing_facilities/
- FIPS Home Page:
  http://csrc.nist.gov/groups/STM/cmvp/index.html
- FIPS Module Validation Lists:
  http://csrc.nist.gov/groups/STM/cmvp/validation.html#02
- NIAP Home Page:
  http://www.niap-ccevs.org/
- NIAP U.S. Government Approved Protection Profiles:
  http://www.niap-ccevs.org/pp/
- NIAP Validated Products List:
  http://www.niap-ccevs.org/vpl/
- Common Criteria Portal (Not controlled by CCEVS):
  http://www.commoncriteriaportal.org/
- DISA STIGs:
  http://iase.disa.mil/stigs/
- Juniper Product Common Criteria Certifications:
  http://pathfinder.juniper.net/compliance/commoncriteria.html
- Juniper Product FIPS Certifications:
  http://pathfinder.juniper.net/compliance/fips.html

Appendix B: Medium Security Sample Configuration

NOTE The following configuration is available on this book's *Day One* landing page at http://www.juniper.net/dayone as a unique file in .rtf format for cutting and pasting into your terminal. The following configuration has extra Return key insertions between the lines to aid in readibility.

```
set version 12.3R9.4

set system host-name MX240

set system time-zone America/New_York

set system default-address-selection

set system no-redirects

set system no-ping-record-route

set system no-ping-time-stamp

set system internet-options tcp-drop-synfin-set

set system internet-options no-source-quench

set system internet-options no-tcp-reset drop-tcp-with-syn-only

set system authentication-order tacplus

set system ports console log-out-on-disconnect

set system ports console insecure

set system ports auxiliary disable

set system ports auxiliary insecure

set system diag-port-authentication encrypted-password ***DISABLED***

set system pic-console-authentication encrypted-password ***DISABLED***

set system root-authentication encrypted-password <PASSWORD>

set system radius-server 192.168.3.20 port 1812

set system radius-server 192.168.3.20 secret <PASSWORD>

set system radius-server 192.168.4.20 port 1812

set system radius-server 192.168.4.20 secret <PASSWORD>

set system tacplus-server 192.168.3.40 port 49

set system tacplus-server 192.168.3.40 secret <PASSWORD>

set system tacplus-server 192.168.4.40 port 49

set system tacplus-server 192.168.4.40 secret <PASSWORD>

set system radius-options password-protocol mschap-v2
```

```
set system accounting events login

set system accounting events change-log

set system accounting events interactive-commands

set system accounting destination radius server 192.168.3.20 accounting-port 1813

set system accounting destination radius server 192.168.3.20 secret <PASSWORD>

set system accounting destination radius server 192.168.4.20 accounting-port 1813

set system accounting destination radius server 192.168.4.20 secret <PASSWORD>

set system accounting destination tacplus server 192.168.3.40 port 49

set system accounting destination tacplus server 192.168.3.40 secret <PASSWORD>

set system accounting destination tacplus server 192.168.4.40 port 49

set system accounting destination tacplus server 192.168.4.40 secret <PASSWORD>

set system login message "\n\tUNAUTHORIZED USE OF THIS SYSTEM\n\tIS STRICTLY PROHIBITED!\n\tPlease
contact company-noc@company.com to gain \n\taccess to this equipment if you need
authorization.\n\n"set system login retry-options tries-before-disconnect 3

set system login retry-options backoff-threshold 1

set system login retry-options backoff-factor 6

set system login retry-options minimum-time 30

set system login retry-options maximum-time 60

set system login retry-options lockout-period 10

set system login class tier1 idle-timeout 10

set system login class tier1 login-alarms

set system login class tier1 login-tip

set system login class tier1 permissions maintenance

set system login class tier1 permissions network

set system login class tier1 permissions view

set system login class tier1 permissions view-configuration

set system login class tier1 deny-commands "(start *)|(set cli idle-timeout)|(request system
software)|(request system zeroize)|(request chassis)"

set system login class tier2 idle-timeout 15

set system login class tier2 login-alarms

set system login class tier2 permissions clear

set system login class tier2 permissions configure

set system login class tier2 permissions interface-control
```

```
set system login class tier2 permissions maintenance

set system login class tier2 permissions network

set system login class tier2 permissions rollback

set system login class tier2 permissions routing-control

set system login class tier2 permissions view

set system login class tier2 permissions view-configuration

set system login class tier2 deny-commands "(start *)|(set cli idle-timeout)|(request system
software)|(request system zeroize)"

set system login class tier2 deny-configuration "(groups)"

set system login class tier3 idle-timeout 20

set system login class tier3 login-alarms

set system login class tier3 permissions all

set system login user emergency full-name "Emergency Only Local Account"

set system login user emergency uid 2010

set system login user emergency class tier3

set system login user emergency authentication encrypted-password <PASSWORD>

set system login user tier1 full-name "Login template for Tier1 Users"

set system login user tier1 uid 2001

set system login user tier1 class tier1

set system login user tier2 full-name "Login template for Tier2 Users"

set system login user tier2 uid 2002

set system login user tier2 class tier2

set system login user tier3 full-name "Login template for Tier3 Users"

set system login user tier3 uid 2003

set system login user tier3 class tier3

set system login password minimum-length 15

set system login password change-type character-sets

set system login password minimum-changes 4

set system login password minimum-numerics 2

set system login password minimum-upper-cases 2

set system login password minimum-lower-cases 2
```

```
set system login password minimum-punctuations 2
set system login password format sha1
set system services ssh root-login deny
set system services ssh no-tcp-forwarding
set system services ssh max-sessions-per-connection 2
set system services ssh ciphers aes256-ctr
set system services ssh ciphers aes256-cbc
set system services ssh ciphers aes192-ctr
set system services ssh ciphers aes192-cbc
set system services ssh ciphers aes128-ctr
set system services ssh ciphers aes128-cbc
set system services ssh macs hmac-sha2-512
set system services ssh macs hmac-sha2-256
set system services ssh macs hmac-sha1
set system services ssh macs hmac-sha1-96
set system services ssh client-alive-count-max 3
set system services ssh client-alive-interval 10
set system services ssh connection-limit 10
set system services ssh rate-limit 2
set system services netconf ssh connection-limit 10
set system services netconf ssh rate-limit 4
set system services web-management https local-certificate <SSL-CERT> set
set system services web-management session idle-timeout 30
set system services web-management session session-limit 4
set system syslog user * any emergency
set system syslog host 192.168.3.2 any any
set system syslog host 192.168.3.2 log-prefix MX240
set system syslog host 192.168.4.2 any any
set system syslog host 192.168.4.2 log-prefix MX240
set system syslog file messages any info
set system syslog file messages authorization info
```

```
set system syslog file User-Auth authorization any

set system syslog file User-Auth interactive-commands any

set system syslog file audit interactive-commands any

set system syslog file processes daemon any

set system syslog console any any

set system syslog time-format year

set system syslog time-format millisecond

set system archival configuration transfer-on-commit

set system archival configuration archive-sites "scp://<USER>@<ADDRESS>:/Configs" password
<PASSWORD>

set system ntp boot-server 192.168.3.2

set system ntp authentication-key 1 type md5

set system ntp authentication-key 1 value <PASSWORD>

set system ntp server 192.168.3.2 key 1

set system ntp server 192.168.3.2 prefer

set system ntp server 192.168.33.2 key 1

set system ntp trusted-key 1

set interfaces ge-0/0/4 description ---unused---

set interfaces ge-0/0/4 disable

set interfaces fxp0 unit 0 description "OOB Management"

set interfaces fxp0 unit 0 family inet address 172.25.46.170/24

set interfaces lo0 unit 0 family inet filter input protect-re

set snmp location DC1-Rack:8-Row:2

set snmp contact "CompanyName NOC:123.456.7890"

set snmp v3 usm local-engine user nms-user authentication-sha authentication-key <PASSWORD>

set snmp v3 usm local-engine user nms-user privacy-aes128 privacy-key <PASSWORD>

set snmp v3 vacm security-to-group security-model usm security-name nms-user group inventory-view

set snmp v3 vacm access group inventory default-context-prefix security-model usm security-level
privacy read-view inventory-view

set snmp v3 vacm access group inventory default-context-prefix security-model usm security-level
privacy notify-view inventory-view

set snmp v3 target-address nms1 address 192.168.3.2

set snmp v3 target-address nms1 tag-list chassis-trap-receivers
```

```
set snmp v3 target-address nms1 target-parameters noc-snmpv3-settings
set snmp v3 target-parameters noc-snmpv3-settings parameters message-processing-model v3
set snmp v3 target-parameters noc-snmpv3-settings parameters security-model usm
set snmp v3 target-parameters noc-snmpv3-settings parameters security-level privacy
set snmp v3 target-parameters noc-snmpv3-settings parameters security-name nms-user
set snmp v3 target-parameters noc-snmpv3-settings notify-filter chassis-traps
set snmp v3 notify chassis-trap-list type trap
set snmp v3 notify chassis-trap-list tag chassis-trap-receivers
set snmp v3 notify-filter chassis-traps oid jnxChassisOKTraps include
set snmp engine-id use-mac-address
set snmp view inventory-only oid jnxBoxAnatomy include
set snmp view inventory-only oid system include
set snmp view system-level oid jnxBoxAnatomy include
set snmp view system-level oid 1.3.6.1.2.1.2 include
set snmp view system-level oid 1.3.6.1.2.1.14 include
set snmp view system-level oid 1.3.6.1.2.1.15 include
set snmp view limited oid 1.3.6.1.2.1.2 include
set snmp client-list performance 192.168.10.0/28
set snmp client-list performance 192.168.20.0/28
set snmp client-list performance 0.0.0.0/0 restrict
set snmp client-list partner 172.16.1.0/28
set snmp client-list partner 172.16.10.0/28
set snmp client-list partner 0.0.0.0/0 restrict
set snmp community "S8M!y:4b" view inventory-only
set snmp community "S8M!y:4b" authorization read-only
set snmp community "S8M!y:4b" clients 192.168.3.3/32
set snmp community "S8M!y:4b" clients 192.168.33.3/32
set snmp community "S8M!y:4b" clients 0.0.0.0/0 restrict
set snmp community "CfL!d4#2" view system-level
set snmp community "CfL!d4#2" authorization read-only
set snmp community "CfL!d4#2" client-list-name performance
set snmp community "xH#5^Gp9" view limited
```

```
set snmp community "xH#5^Gp9" authorization read-only

set snmp community "xH#5^Gp9" client-list-name partner

set protocols rsvp interface ge-0/0/2.0 authentication-key <PASSWORD>

set protocols bgp group session-to-isp1 type external

set protocols bgp group session-to-isp1 peer-as 65000

set protocols bgp group session-to-isp1 neighbor 192.168.11.1 authentication-key <PASSWORD>

set protocols bgp group session-to-core type internal

set protocols bgp group session-to-core local-address 10.10.10.170

set protocols bgp group session-to-core neighbor 10.10.10.86 authentication-algorithm md5

set protocols bgp group session-to-core neighbor 10.10.10.86 authentication-key-chain core-bgp-
keychain

set protocols isis level 1 authentication-key <PASSWORD>

set protocols isis level 1 authentication-type md5

set protocols isis interface ge-0/0/1.0 level 2 disable

set protocols isis interface ge-0/0/1.0 level 1 hello-authentication-key <PASSWORD>

set protocols isis interface ge-0/0/1.0 level 1 hello-authentication-type md5

set protocols isis interface lo0.0 passive

set protocols ospf export advertise-static

set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 authentication md5 1 key <PASSWORD>

set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 authentication md5 2 key <PASSWORD>

set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 authentication md5 2 start-time "2011-3-
31.16:32:00 -0400"

set protocols ospf3 export advertise-static

set protocols ospf3 area 0.0.0.0 interface lo0.0 passive

set protocols ospf3 area 0.0.0.0 interface ge-0/0/1.0 ipsec-sa ospf3-auth-core

set protocols ldp session 10.10.10.85 authentication-key <PASSWORD>

set protocols rip authentication-type md5

set protocols rip authentication-key <PASSWORD>

set protocols rip group eng-group export advertise-static

set protocols rip group eng-group neighbor ge-0/0/1.0 authentication-type md5

set protocols rip group eng-group neighbor ge-0/0/1.0 authentication-key <PASSWORD>

set protocols lldp interface all disable
```

```
set protocols lldp interface ge-0/0/0

set protocols lldp interface ge-0/0/3

set policy-options prefix-list bgp-neighbors apply-path "protocols bgp group <*> neighbor <*>"

set policy-options prefix-list ipv4-interfaces apply-path "interfaces <*> unit <*> family inet
address <*>"

set policy-options prefix-list ospf-all-routers 224.0.0.5/32

set policy-options prefix-list ospf-all-routers 224.0.0.6/32

set policy-options prefix-list ntp-servers apply-path "system ntp server <*>"

set policy-options prefix-list snmp-servers apply-path "snmp community <*> clients <*>"

set policy-options prefix-list mgmt-nets 192.168.3.0/24

set policy-options prefix-list mgmt-nets 192.168.33.0/24

set policy-options prefix-list radius-servers apply-path "system radius-server <*>"

set policy-options prefix-list localhost 127.0.0.1/32

set security certificates local ssl-cert <CERTIFICATE>

set security ipsec security-association ospf3-auth-core description ospf3-neighbor-auth-core

set security ipsec security-association ospf3-auth-core mode transport

set security ipsec security-association ospf3-auth-core manual direction bidirectional protocol ah

set security ipsec security-association ospf3-auth-core manual direction bidirectional spi 256

set security ipsec security-association ospf3-auth-core manual direction bidirectional
authentication algorithm hmac-md5-96

set security ipsec security-association ospf3-auth-core manual direction bidirectional
authentication key ascii-text <PASSWORD>

set security authentication-key-chains key-chain core-bgp-keychain key 0 secret <PASSWORD>

set security authentication-key-chains key-chain core-bgp-keychain key 0 start-time "2011-4-
1.00:01:00 -0400"

set security authentication-key-chains key-chain core-bgp-keychain key 1 secret <PASSWORD>

set security authentication-key-chains key-chain core-bgp-keychain key 1 start-time "2011-7-
1.00:01:00 -0400"

set firewall family inet filter protect-re term synflood-protect from source-prefix-list bgp-
neighbors

set firewall family inet filter protect-re term synflood-protect from source-prefix-list mgmt-nets

set firewall family inet filter protect-re term synflood-protect from protocol tcp

set firewall family inet filter protect-re term synflood-protect from tcp-flags "(syn & !ack) | fin
| rst"

set firewall family inet filter protect-re term synflood-protect then policer limit-100k
```

```
set firewall family inet filter protect-re term synflood-protect then accept

set firewall family inet filter protect-re term allow-bgp from source-prefix-list bgp-neighbors

set firewall family inet filter protect-re term allow-bgp from destination-prefix-list ipv4-
interfaces

set firewall family inet filter protect-re term allow-bgp from protocol tcp

set firewall family inet filter protect-re term allow-bgp from destination-port bgp

set firewall family inet filter protect-re term allow-bgp then accept

set firewall family inet filter protect-re term allow-ospf from source-prefix-list ipv4-interfaces

set firewall family inet filter protect-re term allow-ospf from destination-prefix-list ospf-all-
routers

set firewall family inet filter protect-re term allow-ospf from destination-prefix-list ipv4-
interfaces

set firewall family inet filter protect-re term allow-ospf from protocol ospf

set firewall family inet filter protect-re term allow-ospf then accept

set firewall family inet filter protect-re term allow-ssh from source-prefix-list mgmt-nets

set firewall family inet filter protect-re term allow-ssh from protocol tcp

set firewall family inet filter protect-re term allow-ssh from destination-port ssh

set firewall family inet filter protect-re term allow-ssh then policer limit-10m

set firewall family inet filter protect-re term allow-ssh then accept

set firewall family inet filter protect-re term allow-snmp from source-prefix-list snmp-servers

set firewall family inet filter protect-re term allow-snmp from protocol udp

set firewall family inet filter protect-re term allow-snmp from destination-port snmp

set firewall family inet filter protect-re term allow-snmp then policer limit-1m

set firewall family inet filter protect-re term allow-snmp then accept

set firewall family inet filter protect-re term allow-ntp from source-prefix-list ntp-servers

set firewall family inet filter protect-re term allow-ntp from source-prefix-list localhost

set firewall family inet filter protect-re term allow-ntp from protocol udp

set firewall family inet filter protect-re term allow-ntp from destination-port ntp

set firewall family inet filter protect-re term allow-ntp then policer limit-32k

set firewall family inet filter protect-re term allow-ntp then accept

set firewall family inet filter protect-re term allow-radius from source-prefix-list radius-
servers

set firewall family inet filter protect-re term allow-radius from protocol udp

set firewall family inet filter protect-re term allow-radius from source-port radius
```

```
set firewall family inet filter protect-re term allow-radius from source-port radacct
set firewall family inet filter protect-re term allow-radius then policer limit-32k
set firewall family inet filter protect-re term allow-radius then accept
set firewall family inet filter protect-re term icmp-frags from is-fragment
set firewall family inet filter protect-re term icmp-frags from protocol icmp
set firewall family inet filter protect-re term icmp-frags then syslog
set firewall family inet filter protect-re term icmp-frags then discard
set firewall family inet filter protect-re term allow-icmp from protocol icmp
set firewall family inet filter protect-re term allow-icmp from icmp-type echo-request
set firewall family inet filter protect-re term allow-icmp from icmp-type echo-reply
set firewall family inet filter protect-re term allow-icmp from icmp-type unreachable
set firewall family inet filter protect-re term allow-icmp from icmp-type time-exceeded
set firewall family inet filter protect-re term allow-icmp then policer limit-1m
set firewall family inet filter protect-re term allow-icmp then accept
set firewall family inet filter protect-re term allow-traceroute from protocol udp
set firewall family inet filter protect-re term allow-traceroute from destination-port 33434-33523
set firewall family inet filter protect-re term allow-traceroute then policer limit-1m
set firewall family inet filter protect-re term allow-traceroute then accept
set firewall family inet filter protect-re term tcp-established from protocol tcp
set firewall family inet filter protect-re term tcp-established from source-port ssh
set firewall family inet filter protect-re term tcp-established from source-port bgp
set firewall family inet filter protect-re term tcp-established from tcp-established
set firewall family inet filter protect-re term tcp-established then policer limit-10m
set firewall family inet filter protect-re term tcp-established then accept
set firewall family inet filter protect-re term default-deny then log
set firewall family inet filter protect-re term default-deny then syslog
set firewall family inet filter protect-re term default-deny then discard
set firewall policer limit-10m if-exceeding bandwidth-limit 10m
set firewall policer limit-10m if-exceeding burst-size-limit 625k
set firewall policer limit-10m then discard
set firewall policer limit-3m if-exceeding bandwidth-limit 3m
set firewall policer limit-3m if-exceeding burst-size-limit 15k
```

```
set firewall policer limit-3m then discard
set firewall policer limit-1m if-exceeding bandwidth-limit 1m
set firewall policer limit-1m if-exceeding burst-size-limit 15k
set firewall policer limit-1m then discard
set firewall policer limit-100k if-exceeding bandwidth-limit 100k
set firewall policer limit-100k if-exceeding burst-size-limit 15k
set firewall policer limit-100k then discard
set firewall policer limit-32k if-exceeding bandwidth-limit 32k
set firewall policer limit-32k if-exceeding burst-size-limit 15k
set firewall policer limit-32k then discard
```

Appendix C: Hardening Junos Devices Checklist

The last two pages of this book are a sample checklist that you can print and use in your own lab or production networks. The checklist also exists as a separate PDF you can freely download on this book's *Day One* landing page at: http://www.juniper.net/dayone.

Hardening Junos Devices Checklist

The companion checklist to *This Week: Hardening Junos Devices, Second Edition*

Date: _____ Device Name: _____ Location: _____ Rack/Row: _____

IP Address: _____ NetMask: _____ Gateway: _____ MAC: _____

Administrative (see Chapter 1)

- ☐ Research the latest Juniper Security Advisories
- ☐ Install recommended version of Junos: _____

Physical Security (see Chapter 2)

- ☐ If redeploying a previously installed device, perform a media installation to remove previous configurations and data

Secure Physical Ports

- ☐ Disable unused network ports

Console Port

- ☐ Configure the logout-on-disconnect feature
- ☐ Configure the insecure feature

Auxiliary Port

- ☐ Disable the Auxiliary port
- ☐ Configure the insecure feature

Diagnostic Ports

- ☐ Password protect Diagnostic ports

Craft Interface/LCD Menu

- ☐ Disable unnecessary functions for your environment

Network Security (see Chapters 3 & 4)

- ☐ Use the Out-of-Band (OOB) interface for all management related traffic (Ch. 3)
- ☐ Enable the default-address-selection option (Ch. 4). Set the source address for all route engine generated traffic (NTP, SNMP, Syslog, etc.)
- ☐ Globally disable ICMP redirects (Ch. 4)
- ☐ Ensure Source Routing has not been configured (Ch. 3)
- ☐ Ensure IP directed broadcast has not been configured (Ch. 3)
- ☐ Ensure Proxy ARP is either not configured, or is restricted to specific interfaces (Ch. 3)
- ☐ Drop TCP packets with the SYN and FIN flag combination (Ch. 4)
- ☐ Disable ICMP timestamp & record route requests (Ch. 4)
- ☐ Disable ICMP Source Quench
- ☐ Configure LLDP only on required network ports (Ch. 4)

Management Services Security (see Chapter 4)

- ☐ Configure NTP with authentication with more than one trusted server
- ☐ Configure SNMP using the most secure method with more than one trusted server
- ☐ Community strings and USM passwords should be difficult to guess and should follow a password complexity policy
- ☐ Configure read-only access; use read-write only when required
- ☐ Allow SNMP queries and/or send traps to more than one trusted server
- ☐ Send Syslog messages to more than one trusted server with enhanced timestamps
- ☐ Configure automated secure configuration backups to more than one trusted server

Access Security (see Chapter 4)

- ☐ Configure a warning banner that is displayed prior to login
- ☐ Disable insecure or unnecessary access services (telnet, J-Web over HTTP, FTP, etc.)
- ☐ Enable required secure access services:

SSH

- ☐ Use SSH version 2
- ☐ Deny Root logins
- ☐ Set connection-limit and rate-limit restrictions

J-Web

- ☐ Use HTTPS with a valid certificate signed by a trusted CA
- ☐ Limit access to only authorized interfaces
- ☐ Terminate idle connections by setting the idle-time value
- ☐ Set session-limit restrictions suitable for your environment

Continued on Page 2

User Authentication Security (see Chapter 4)

- ☐ Configure a password complexity policy
  - ☐ Minimum password length, upper case, lower case and special characters
  - ☐ Use SHA1 for password storage
- ☐ Ensure the root account has been configured with a strong password
- ☐ Configure login security options to hinder password guessing attacks
- ☐ Configure custom login classes to support engineers with different access levels using the least privilege principle
  - ☐ Restrict commands by job function
  - ☐ Set appropriate idle timeout values for all login classes
  - ☐ Limit access to ## SECRET-DATA

Centralized authentication

- ☐ Use a strong shared secret that complies with your organization's password complexity policy
- ☐ Configure multiple servers for resiliency
- ☐ Configure accounting to trace activity and usage
- ☐ Create an emergency local account in the event authentication servers are unavailable

Local Authentication

- ☐ Know the origin and purpose for all configured local accounts
- ☐ Limit local accounts to required users
- ☐ Use a strong password that complies with your organization's password complexity policy
- ☐ Set the authentication-order to meet your login security policy

Routing Protocol Security (see Chapter 4)

- ☐ Ensure routing protocols are only configured on required interfaces
- ☐ BGP communication should source from a loopback interface
- ☐ Configure route authentication with internal and external trusted sources
  - ☐ Select the strongest algorithm that is supported by your equipment and your neighbors
  - ☐ Use strong authentication keys that meet your organization's password complexity policy
  - ☐ Limit key exposure by using separate authentication keys for different organizations
- ☐ Periodically change route authentication keys in accordance with your organization's security policy (consider using hitless key rollover if the routing protocol supports it)

Firewall Filter (see Chapter 4)

- ☐ Protect the Routing Engine using a default deny firewall filter
- ☐ Order terms with time sensitive protocols at the top
- ☐ Permit only required protocols from authorized sources
- ☐ Rate-limit SYN packets to protect against a SYN flood attack
- ☐ Rate-limit authorized protocols using policers
- ☐ Ensure the last term, default-deny, includes the syslog option

Installer: _____ Installer Phone: _____ Installer Email: _____

Owner: _____ Owner Phone: _____ Owner Email: _____

This excerpt is from *This Week: Hardening Junos Devices, Second Edition*, available at http://www.juniper.net/dayone, and also available in eBook format on the iTunes Store>iBooks or the Amazon.com Kindle store.

THIS WEEK COMPANION

Hardening Junos Devices Checklist

Juniper Networks Information and Learning Experience (iLX)

www.juniper.net/posters