Juniper Networks Secure Development Lifecycle

Six Practices for Improving Product Security

Table of Contents

List of Figures

Executive Summary

This document describes Juniper Networks' process for developing products that are more secure, more resilient, and better able to address security requirements for service provider customers, enterprise customers, and Internet users. It addresses customer requirements for a security assurance process that works to improve software security while conforming to industry practices. This document outlines six separate security activities, or practices, that work together to enhance the process for developing secure products. Finally, it establishes these enhanced practices as an integral security layer within Juniper Networks' product development methodology.

Introduction

> **"Invisible and essential as air, the global Internet is the most important engineering feat in modern history."**
>
> Scientific American, September 2012

The Internet is the defining accomplishment of our generation, and Juniper Networks contributes to this amazingly powerful system through its extensive portfolio of networking products. Juniper products and services are critical for maintaining the global flow of information across network core, access, and aggregation layers. Juniper products power the world's most demanding networks, including top global service providers and 99 of the Fortune Global 100 companies.

For many companies, concerns over security and the need for secure products are at the center of today's business drivers:

- Business survival: Company reputation and future business growth rely on having secure products
- Threats: Attacks are more sophisticated, with an increasing focus on monetary gain and espionage
- Technology: Sophisticated hacking tools are being developed by organized entities
- Regulatory matters: Government, industry, and customer contracts require security assurances

Internet end users, along with enterprise and service provider customers, are depending on Juniper to do its part to ensure product security. Malicious entities are continually seeking to disrupt the Internet, break its universal connectivity, or breach its integrity. Juniper products must be resilient enough to withstand these attacks and keep the Internet running.

Given this critical role, Juniper has implemented the Juniper Networks Secure Development Lifecycle.

Juniper Networks Secure Development Lifecycle

Juniper's Secure Development Lifecycle methodology consists of a sequence of six product security activities designed to provide assurance that Juniper products are sufficiently secure and resilient. While individually each practice addresses aspects of secure software development, together these practices create a layered and holistic security model that works in conjunction with Juniper's product development processes. The six Secure Development Lifecycle practices cover a range of security functions:

- Practice #1: Secure Coding Training
- Practice #2: Security Consideration in Design
- Practice #3: Threat Modeling
- Practice #4: Penetration Testing
- Practice #5: Release Security Review
- Practice #6: Incident Response Plan

Figure 1 illustrates the serial nature of Secure Development Lifecycle practices while still providing for information exchange between any of the practices.

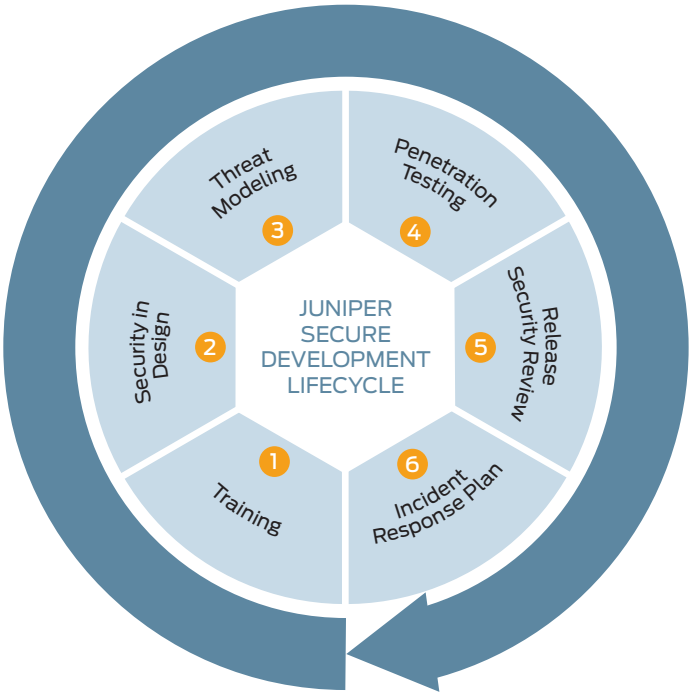

Figure 1: Juniper Networks Secure Development Lifecycle practices

Each of these practices has a role to play in managing and mitigating risks due to product development security issues that could impact Juniper's business, customers, and Internet users. By adopting this methodology, Juniper assesses the baseline security of products and takes active steps to help customers maintain or improve their security posture as new products are introduced and their business grows.

Secure Development Lifecycle practices align with Juniper's overall Product Development Lifecycle (PDL) methodology, which provides a consistent process for product planning, design, implementation, test, release, and on-going support.

Conceptually, the Secure Development Lifecycle adds a lightweight process layer to the PDL. Information involved in secure product development not only flows between practices, but is also exchanged with processes related to the overall product release and delivery in the PDL.

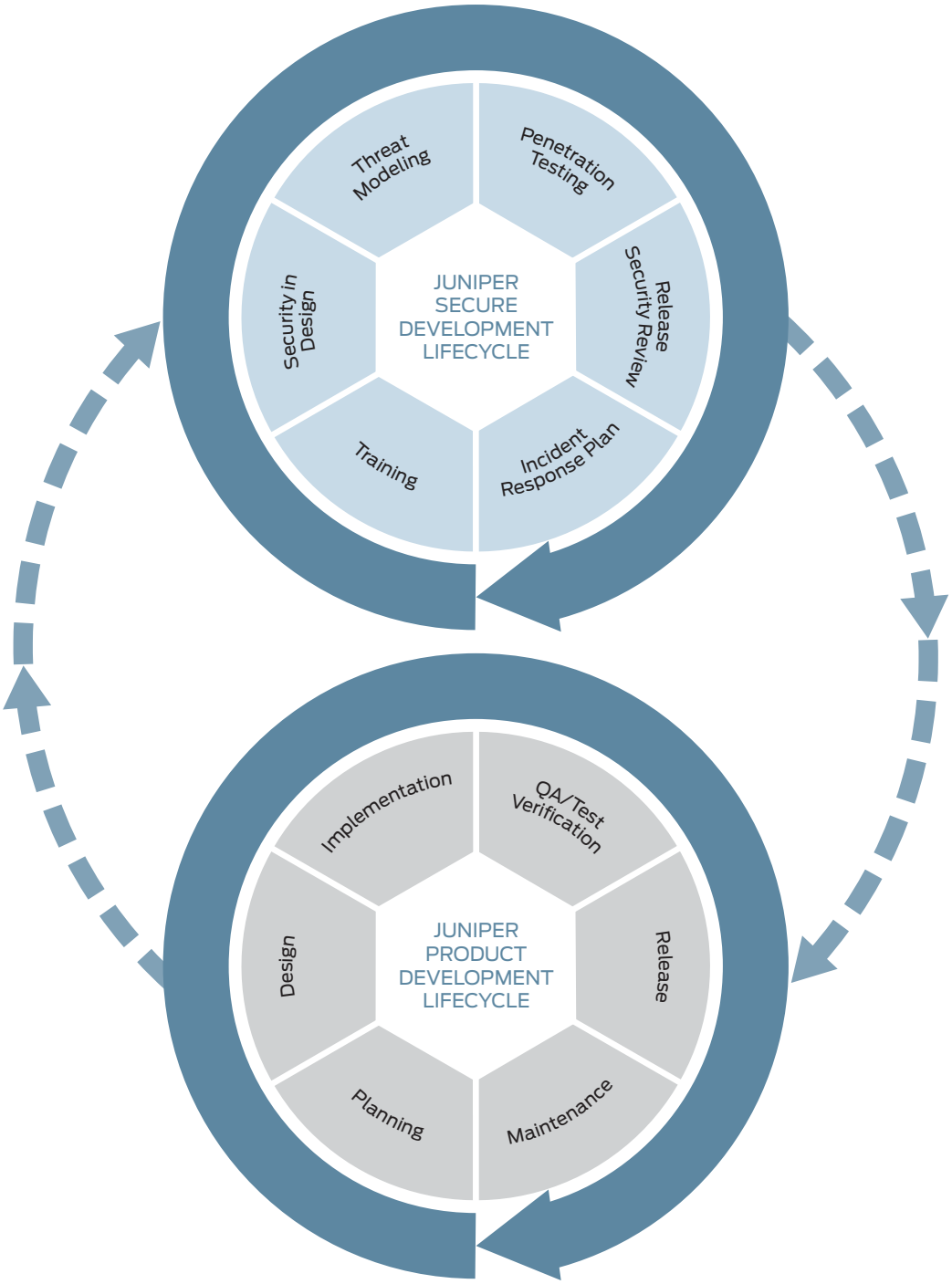

Figure 2: Secure Development Lifecycle lightweight process

While Secure Development Lifecycle practices generally flow sequentially in conjunction with PDL phases, the methodology is flexible enough to address security concerns independent of product or feature lifecycles. In other words, output from any one of the Secure Development Lifecycle practices can serve as input to one or more other practices. For example, feedback from the Incident Response Plan can effect changes in Secure Coding Training or Threat Modeling for a product. At the same time, flaws uncovered during Penetration Testing might be flagged for special consideration during Release Security Review.

Ultimately, the six practices that comprise Juniper's Secure Development Lifecycle work to address the five goals of Internet security:

- **Confidentiality**—ensuring that only authorized individuals have access to the resources being exchanged
- **Integrity**—guaranteeing that the message sent is the message received
- **Availability**—ensuring the information system's proper operation
- **Non-repudiation**—guaranteeing that an operation cannot be denied
- **Authentication**—ensuring that only authorized individuals have access to the resources

The following sections define each of the Secure Development Lifecycle practices.

Practice #1: Secure Coding Training

At the heart of Juniper's commitment to making its products more secure is a commitment to training our engineering community in secure coding techniques. Secure Coding Training is a prerequisite for implementing the Secure Development Lifecycle. All software developers at Juniper are required to take the training, which is foundational for building more resilient software. Training is provided in multiple coding languages, with developers taking the appropriate course: C/C++, Java, .NET, PHP, and so on.

Secure Coding Training covers fundamental concepts related to secure coding, secure design, secure testing, and privacy, including the following topics:

Secure Coding

- Buffer overflows
- Integer arithmetic errors
- Input and output validation
- Handling errors with return codes
- Cross-site scripting
- Weak cryptography

Secure Design

- Principle of least privilege
- Data access threats and defenses
- Secure defaults
- Security testing
- Verification techniques
- Code review
- Static and dynamic analysis

Privacy

- Types of private information
- Privacy management
- Cryptography, passwords, and secrets

While Secure Coding Training is mandatory for all developers, Juniper believes that everyone involved in software development is responsible for security in the product software. This includes managers, program managers, testers, and IT personnel. With this in mind, Secure Development Lifecycle training is available 24 hours a day, 7 days a week to all employees, and it offers a range of additional training covering secure coding fundamentals:

- Security testing principles
- Application security
- IT security

Building a baseline of secure coding knowledge and awareness among all those involved in software development benefits the overall implementation of the Secure Development Lifecycle. And it also helps in facilitating each individual practice, especially when it comes to evaluating security considerations in the functionality design.

Practice #2: Security Considerations in Design

Addressing security is important at every stage of product development, but it is especially critical to address security at the beginning of a project through careful consideration and planning. Identifying and fixing security issues at the end of a product development cycle or after a product release can be expensive, both in terms of the rework cost and with regard to schedules, deliverables, revenue, and customer good will. Security risks not only come from determined attackers, but also from innocent oversights in software administration.

When developers consider security in the functional design, they conduct three basic tasks:

- Evaluate for vulnerabilities
- Assess the possibility of a threat compromising a vulnerability
- Mitigate the vulnerability in design or recommend suitable secure deployment of functionality

Reviewing the vulnerabilities and threats that might arise in software, along with the associated mitigations relative to the operation of the product, establishes a security posture, or baseline security state for the product. You can think of this in the same way that a military organization evaluates, anticipates, and prepares to address security issues, thereby establishing a posture relative to a specific threat or vulnerability. The security posture for a router deployed in a bank is different from that of a switch installed in an office.

Juniper's PDL process states that functional specifications associated with every new Juniper Networks® Junos® operating system feature must have a security considerations section, which evaluates threats on a specific feature, protocol, or technology. The types of attacks and threats associated with the software security posture depend on the product and can include, but are not limited to, eavesdropping, replay, message insertion, message deletion, modification of data, man-in-the-middle, and denial of service (DoS). Some products might also require a review of cryptographic functions.

Reviewing security relative to feature functionality is accomplished through a series of Secure Development Lifecycle activities:

- Vulnerability assessments using industry standard information and publications such as RFCs
- Security posture evaluation in product software functional specifications
- Security feedback integrated into product planning and design

A solid understanding of a product's security posture through review of the functionality design serves as input into other Secure Development Lifecycle practices, including Threat Modeling.

Practice #3: Threat Modeling

Many products on the market today provide security functionality, but providing security in a product does not necessarily mean the product is secure. Building a secure product means you have worked to consider and understand the security risks and vulnerabilities associated with that product. Vulnerabilities, by definition, are not always accounted for by developers and therefore can become risks. Additionally, attackers think differently than developers. Attackers will not use the product as it was designed and will look for any means by which to exploit vulnerabilities in the functionality. To truly build a more secure product, developers must understand the threats to a product.

Threat Modeling is a process that evaluates the product to identify potential threats that could compromise it, or potential threats that aid in compromising other components or systems interacting with that product. Threat Modeling determines risks from those threats and sets the boundaries for a range of appropriate mitigations.

The threat model also helps developers to define a product's attack surfaces, meaning the breadth and depth of exposure to compromise. For example, a weak password can be exploited by a brute force attack, or the use of a predictable TCP/IP ephemeral port may allow an attacker to mount a TCP reset attack.

Threat Modeling builds a framework for deeper security evaluation by identifying and enumerating issues:

- Identifies and enumerates security assumptions based on the product security posture and functional specifications
- Identifies and enumerates security risks in feedback from other Secure Development Lifecycle practices
- Conducts system decomposition, identifying the actors, processes, data flows, data stores, and trust boundaries
- Identifies threats and vulnerabilities
- Enumerates threats using industry standard Common Vulnerabilities and Exposures (CVE) identifiers

Figure 3 represents a threat model for an application running in a networked environment, and illustrates how developers relate elements of the threat model when evaluating product security.

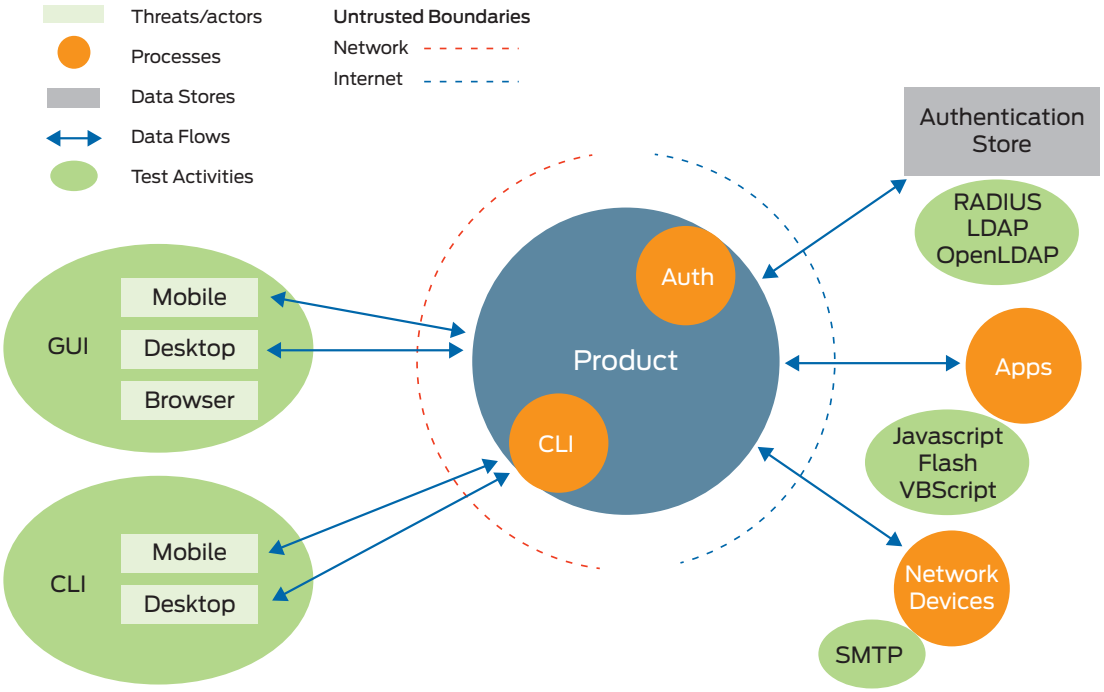

Figure 3: Example threat model for an application running in a networked environment

The output of Threat Modeling serves as input to other phases of the Secure Development Lifecycle, including Penetration Testing.

Practice #4: Penetration Testing

With a product's security posture defined and threat model documented, the Secure Development Lifecycle calls for evaluation and validation of the security risks through "ethical hacking" of identified attack surfaces. The most common method of validation is Penetration Testing. Penetration Testing is a security evaluation methodology in which ethical hackers mimic real-world attacks to identify methods for circumventing the security features of an application, system, or network. It involves launching real attacks on test systems, using tools and techniques commonly used by adversaries.

Penetration Testing makes use of the threat model to devise a penetration test plan based on enumerated attack surfaces and threats. The threat model illustrated in Figure 3 demonstrates how threat model information is used in penetration test planning, identifying facets of various attack surfaces and listing associated test points.

The penetration test plan describes tools and techniques to be employed and includes a combination of manual and automated tests. Penetration Testing generates exploits that can compromise a vulnerability in the product and breach the confidentiality, integrity, or availability of the system. The key strategies of Penetration Testing generally focus on high-risk, high-threat components comprising the product. The types of tests depend on what attack surfaces were identified in the threat model. For example, Penetration Testing for web-based applications might include access from a mobile device, whereas a switch in a wiring closet might not have the same access.

Penetration Testing identifies potential vulnerabilities. Areas of focus for testing can include, but are not limited to, authentication, authorization, cryptography, and access privileges. Vulnerabilities identified through Penetration Testing can be fixed or mitigated in the PDL prior to the product release. Vulnerabilities learned from penetration tests provide a baseline for future product testing.

As with all other practices forming the Juniper's Secure Development Lifecycle, lessons learned in Penetration Testing can provide feedback to other practices. For example, bugs found during Penetration Testing can serve as input to Security Considerations in Design, and as input to the Release Security Review.

Practice #5: Release Security Review

With each practice, the Secure Development Lifecycle is building a picture of a product's security posture. Security considerations defined in the functional specifications, threat conditions identified through modeling, and penetration test results all provide a piece of the security picture. The Release Security Review is a deliberate examination of a product's security posture prior to release with the goal of identifying and evaluating the security risks, which might then be eliminated, reduced, mitigated or accepted in relation to the PDL.

The Release Security Review considers the following factors:

- Background information about the product and related documentation
- Secure Coding Training requirements
- Functional specifications and any recommendations for product enhancements
- Threat Modeling analysis results
- Penetration Testing results and the number and types of issues logged
- Bug reviews, including the number and priority
- Prior product security reviews

From this information, the security team can evaluate the severity of the issues reported in all Secure Development Lifecycle practices and work to fix those issues prior to release of the product.

As well as defining the overall security posture for the product, results of the Release Security Review also serve as input to other Secure Development Lifecycle practices.

Practice #6: Incident Response Plan

Network technologies today are complex, and individual features can comprise hundreds of files, thousands of objects, and millions of lines of code. Also, products, systems, and solutions are becoming more interdependent. A product that was released with no known vulnerabilities can become subject to threats over time.

The Incident Response Plan outlines how Juniper responds to potential product vulnerabilities and how these threats and mitigations are communicated to customers. This practice builds on Juniper's industry-respected Security Incident Response Team (SIRT) framework for responding to security issues. In responding to security incidents, the plan relies on existing SIRT tools, best practices, processes, and relationships:

- Dedicated security experts available for response
- Potential vulnerabilities identified from multiple sources, including internal review, customer service, threat monitoring, and cooperation with researchers, partners, and other vendors
- Security bugs evaluated and scored using industry standard Common Vulnerability Scoring System (CVSS) methodology
- Customer advisories published with industry standard Common Vulnerabilities and Exposure (CVE) identifiers to aid customer understanding
- Established communications channels for resolving incidents
- Information sharing with industry peers to prevent the spread of threats and allow for fixing of vulnerabilities in multivendor customer networks

Threats and vulnerabilities identified through the Incident Response Plan are included in the Release Security Review for consideration of the overall product security posture.

Conclusion

Juniper Networks Secure Development Lifecycle is a series of six security practices implemented in conjunction with Juniper's product development methodology to create an environment where secure coding and product development go hand-in-hand. This environment provides a framework for building products that are more secure and resilient to the ever changing landscape of Internet threats, and that meet the ever growing need for secure products that address the business drivers of companies both today and into the future. From secure coding practices to responding to security incidents, the Secure Development Lifecycle works to secure products against entities intent on disrupting the flow of information across the Internet, launching malicious attacks, or engaging in espionage.

Given the critical role Juniper products play in maintaining the global flow of information, Internet users, as well as companies doing business around the world, depend on Juniper Networks to do all it can to build secure products. The Secure Development Lifecycle is one way Juniper Networks is leading the way.

For more information about the Secure Development Lifecycle, please contact sdl-feedback@juniper.net.

About Juniper Networks

Juniper Networks is in the business of network innovation. From devices to data centers, from consumers to cloud providers, Juniper Networks delivers the software, silicon and systems that transform the experience and economics of networking. The company serves customers and partners worldwide. Additional information can be found at www.juniper.net.

Corporate and Sales Headquarters
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089 USA
Phone: 888.JUNIPER (888.586.4737)
or +1.408.745.2000
Fax: +1.408.745.2100
www.juniper.net

APAC and EMEA Headquarters
Juniper Networks International B.V.
Boeing Avenue 240
1119 PZ Schiphol-Rijk
Amsterdam, The Netherlands
Phone: +31.0.207.125.700
Fax: +31.0.207.125.701

2000526-001-EN Sept 2015