OIC-CERT Software Supply Chain Security Framework

Version 2.0

Document Date: 20 December 2024

© OIC-CERT Software Supply Chain Security Framework 2024

All rights reserved. Unless otherwise specified or required in the course of implementation, not in any form or by any means without prior written permission (Electronic or mechanical, including photocopying or posting on the Internet or intranet) Reproduction or use of any part of this publication. You can apply for a license from the OIC-CERT Permanent Secretariat at the following address.

Standard office name: OIC-CERT Permanent Secretariat, CyberSecurity Malaysia
Address: Level 4, Tower 1, Menara Cyber Axis, Jalan Impact, 63000 Cyberjaya, Selangor, Malaysia
Phone: +603-8800 7999
E-mail: secretariat@oic-cert.org
Website: www.oic-cert.org

Published in www.oic-cert.org

Contents

Audience

The framework is aimed primarily at the regulatory authorities of member countries and is intended to assist them in the formulation of policies for supply chain manufacturers and related service providers.

1 Software Supply Chain Security Overview

1.1 The Internet of Everything (IoE) era poses challenges to software supply chain security.

Today, with the wide application of technologies such as mobile Internet, artificial intelligence, big data, cloud computing, blockchain, and quantum computing, we have gradually entered a new digital era. Digital technologies touch all aspects of people's lives and work. Its openness and connectivity are also changing the way people live and work around the world. Data has become the basic resource and production factor for modern industry development. Every country is issuing laws and regulations to protect data sovereignty and security. As an important carrier of digital technology, software application system plays an important basic support role in the digital era. At present, the software industry is developing rapidly. Basic software, industrial software, system software, and application software constitute a huge software system in enterprise information systems. Secure software and information systems have become the premise and foundation for enterprises to protect the security of data and information assets. As software systems become more and more complex and integrated, the security of third-party purchased software and open source software is difficult to ensure. As a result, security incidents occur frequently in the current software supply chain. How to establish a sound software supply chain security system has become a problem that global enterprises must face.

When we look back at the stage of scientific and technological development, we can find that society's demand for information and data security existed long ago. It is said that in ancient Rome, there was a rudiment of information encryption, and Caesar used letter offset encryption to make military contact with his generals. After that, in the modern stage, especially with the development of telegraph and telephone technology, information encryption and information security in the communication process have become the main concern of the society. After World War II, network technology and computer technology have developed by leaps and bounds. After the wide application of the Internet, larger-scale network attacks, information leakage, and data abuse have emerged, further intensifying the challenges facing network security.

With the emergence of AIGC and breakthroughs in big data and IoT technologies, we can foresee that while technology drives the prosperity of the digital economy, society's requirements for cyber security, data security, and privacy protection are also evolving. With the globalization of the supply chain, software and hardware products become more and more complex. Strengthening software supply chain security is a very important part of ensuring the prosperity and development of the digital economy. However, software supply chain attacks are constantly evolving, from software build environment attacks (SolarWinds) to software supply chain attacks (Log4j), which is a very natural attack evolution process and poses a challenge to software supply chain security.

1.2 Software Supply Chain Security Concept

In the brand-new digital era, digital software applications, as the key carrier of information technology, drive the rapid development of digital industry and become the basic elements of people's production and life. Online shopping, digital wallet, social platform, short video, and live broadcast have become important parts of people's life. Therefore, while accelerating the development of the digital economy, effectively ensuring the security of the software supply chain can better enable people's production and life and lay a solid foundation for the development of new technologies such as artificial intelligence.

Nowadays, the development mode of software applications has changed from the original iterative development mode to the agile development mode, especially the arrival of the open source era, which has reshaped the software development ecosystem and changed the software development mode. The open source model of equality, openness, collaboration, and sharing is accelerating the iterative upgrade of software. Open source has become the dominant model of global software technology and industry innovation, and is the cornerstone of technology application and industry digital development. Open source plays an important role in promoting software innovation, promoting technology development and building open cooperation between enterprises. According to Gartner's survey, 99% of organizations use open source components in their information systems, and open source software and components are the main components of software today. The use of open source and the diversity of software sources make the security problems of software supply chain increasingly prominent. To solve the security problems faced by software supply chain effectively, it is necessary to clarify the relevant concepts and definitions of software supply chain security.

Software supply chain: Often defined as the organization, people, activities, information, and resource systems involved from development to delivery of products or services to customers. Components of the software supply chain include suppliers, developers, repositories, dependent components, tools, and end users.

Software supply chain security: Software supply chain security refers to the security of code, tools, devices, and upstream code, modules, and services in each phase of software design and development in the software supply chain, and the security of software delivery channels. The software supply chain includes software development, testing, packaging, distribution, and deployment, as well as all vendors and third-party organizations involved. Software supply chain security requires a series of measures, such as reviewing the credibility and security of suppliers, implementing code review and vulnerability scanning, using technologies such as digital signature and encryption to ensure the integrity and authenticity of software, and implementing measures such as access control and monitoring to protect software systems from attacks.

Software supply chain attack: A specific threat that directly targets developers and suppliers. An attacker infects legitimate applications, distributes malware, accesses source code, build process, or updates to attack software.

1.3 Typical Security Events of Software Supply Chain

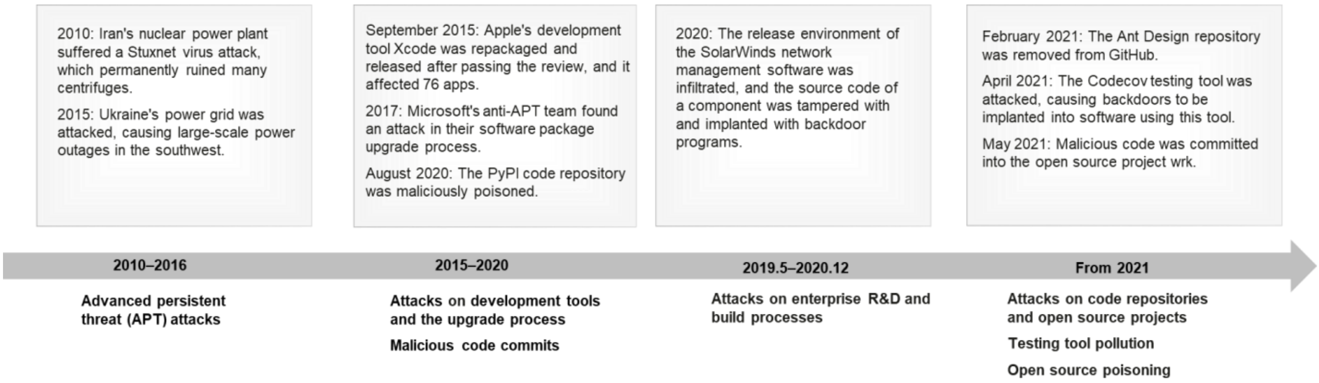
In December 2020, it was disclosed that SolarWinds, the world's most famous network management software supplier, suffered a highly complex supply chain attack by a national-level APT gang and planted a Trojan horse backdoor, SUNBURST.

Hackers have inserted backdoors into the updated version of Orion by breaking into the server SolarWinds used to build the updated Orion version. These versions are posted on the SolarWinds website, and customers download them and load them as normal.

This attack directly affected 18000 enterprise customers using some versions of SolarWinds Orion management software. It is called the most influential supply chain attack in history, affecting a large number of government agencies, enterprises, and organizations, including US government agencies, military institutions, and foreign ministries around the world.

The SolarWinds incident revealed the threat of supply chain attacks, where hackers penetrate the network systems of targeted organizations by attacking a link in the software supply chain. This type of attack is highly covert and makes it difficult for the attacked organization to detect it. At the same time, it has triggered a global cyber security warning and led countries to pay more attention to cyber security.

We can see that over the past decade, cyber security attacks have become increasingly severe, and the software supply chain has faced great challenges:



2010: Iran's nuclear power plant suffered a Stuxnet virus attack, which permanently ruined many centrifuges.

2015: Ukraine's power grid was attacked, causing large-scale power outages in the southwest.

September 2015: Apple's development tool Xcode was repackaged and released after passing the review, and it affected 76 apps.

2017: Microsoft's anti-APT team found an attack in their software package upgrade process.

August 2020: The PyPI code repository was maliciously poisoned.

2020: The release environment of the SolarWinds network management software was infiltrated, and the source code of a component was tampered with and implanted with backdoor programs.

February 2021: The Ant Design repository was removed from GitHub.

April 2021: The Codecov testing tool was attacked, causing backdoors to be implanted into software using this tool.

May 2021: Malicious code was committed into the open source project wrk.

2010–2016

Advanced persistent threat (APT) attacks

2015–2020

Attacks on development tools and the upgrade process

Malicious code commits

2019.5–2020.12

Attacks on enterprise R&D and build processes

From 2021

Attacks on code repositories and open source projects

Testing tool pollution

Open source poisoning

Software supply chain attacks are becoming an increasingly common criminal method for malicious destruction and illegal information acquisition. According to Gartner, a research firm, 45% of enterprises will be attacked by supply chain attacks by 2025. By summarizing the software supply chain security incidents in recent years, we can find that software supply chain attacks are characterized by various attack modes, wide attack areas, large impact scope, strong concealment, and unpredictable attacks. Attackers cover multiple software participants, such as suppliers, users, and developers. Attack methods include poisoning, dependency confusion, malicious code injection, sanctions, interruption of supply, and suspension of service.

1.4 Policies, Regulations, and Standards of Each Country

Faced with the rapidly growing threat to the software supply chain, governments around the world have issued regulations and policies, focusing on software security design, security development, software responsibility and self-certification, and third-party certification.

Table 1 - Major Global Software Supply Chain Security Laws and Regulations Released in Recent Years

United States of America	
Presidential Executive Order on Cybersecurity	President Biden's Executive Order 14028 - sets out guidelines for the National Institute of Standards and Technology (NIST), the Office of Management and Budget (OMB), the Cybersecurity and Infrastructure Security Agency (CISA), and others.
Office of Management and Budget (OMB): Memorandums 22-18 & 23-16	Two memos, 22-18 and 23-16, were issued, each focusing on software supply chain security and starting to push requirements, such as requirements for all software vendors selling to the U.S. federal government. Governments began self-certifying compliance with secure software development practices, such as the NIST Secure Software Development Framework (SSDF). The memo also requires the use of SBOM in some cases and third-party assessment services for higher-risk projects.
NIST National Institute of Standards and Technology: SSDF Secure Software Development Framework	NIST's Secure Software Development Framework (SSDF) is a required course in understanding U.S. software supply chain security. The U.S. President's Executive Order on Cybersecurity explicitly requires NIST to update the SSDF and OMB, which are key frameworks for self-certification for all software vendors selling products to the U.S. federal government. SSDF leverages multiple existing secure software development frameworks, such as OWASP's Secure Application Maturity Model (SAMM) and Synopsys' Build Security Maturity Model (BSIMM), to cross-reference practices when developing secure software.
NCS National Cyber Security Strategy Software Responsibility	A key theme of NCS's strategy is the shift in focus from customers and consumers to software vendors, and this is a key theme of "Security by Design" initiatives by institutions such as CISA. Another key theme for NCS is an emphasis on shaping market forces to drive security and resiliency, and calls for activities such as holding data managers accountable and driving the development of secure devices, even introducing the controversial "software accountability" theme.
Open Source Software Security Act of 2023	Similar to the enterprise market, the U.S. federal government has become increasingly reliant on open source software. This was publicly acknowledged by the Protection of Open Source Software Act of 2022. The bill recognizes the importance of open source software and calls on institutions such as CISA to engage directly with the open source community. It defines the responsibilities of the CISA Director for outreach and engagement to help promote the security of the open source software ecosystem.
European Union	
Cyber Resilience Act	The Act covers hardware and software as well as any products with "digital elements" and is very similar to GDPR. The Act requires cyber security to be a key factor in the design and development of products with digital elements, and violations can lead to restrictions on sales of products in the EU market, in addition to administrative fines.
Artificial Intelligence	The AI Act sets out a variety of acceptable risk levels, from "low," "minimum," to "completely

Act	prohibited certain uses," such as uses that result in violations of human dignity or manipulation of human behavior. The Act applies to all AI systems and services launched and used in the EU, and therefore has a global impact. Producers deemed to be high-risk systems are required to perform a variety of risk management and governance activities and self-certify compliance with the Act, and violations can result in fines of up to 4% of global turnover or tens of millions of Euros.
China	
Regulations on the Security and Protection of Critical Information Infrastructure	Article 19 of the Regulations on the Security Protection of Critical Information Infrastructure clearly states that operators shall give priority to the procurement of secure and reliable network products and services; Where the procurement of network products and services may affect national security, security examination shall be passed in accordance with the national cyber security regulations.
Cyber security review method	The Cyber Security Review Measures clearly state that in order to ensure the security of the supply chain of critical information infrastructure and maintain national security, cyber security reviews should be conducted for network products and services purchased by critical information infrastructure operators that affect or may affect national security.
Canada	
CCCS Changing the Cybersecurity Risk Balance: Principles and Methods for Design and Default Security	The Canadian Cyber Security Centre (CCCS) contributed to the publication of Changing the Balance of Cyber Security Risk: Principles and Methods for Security by Design and Default. It also identified software supply chain attacks as a key issue in the CCCS 2023-2024 National Cyber Threat Assessment.
Australia	
ACSC Software Development Guide	Focus on various security controls across the software development lifecycle and environment. It also highlights the need for application security controls and testing to fix vulnerabilities and cites use cases for SBOM. Australia is also a part of the international project "Quartet Cyber Security Partnership: Joint Principles for Secure Software". Released in May 2023, developed in collaboration with the United States, India, Japan and Australia. The focus is on incorporating secure software development practices into government policies and software procurement by vendors. It aligns with the four phases in the NIST SSDF and discusses the proposal to require software manufacturers to self-certify and even, if necessary, third-party certification.

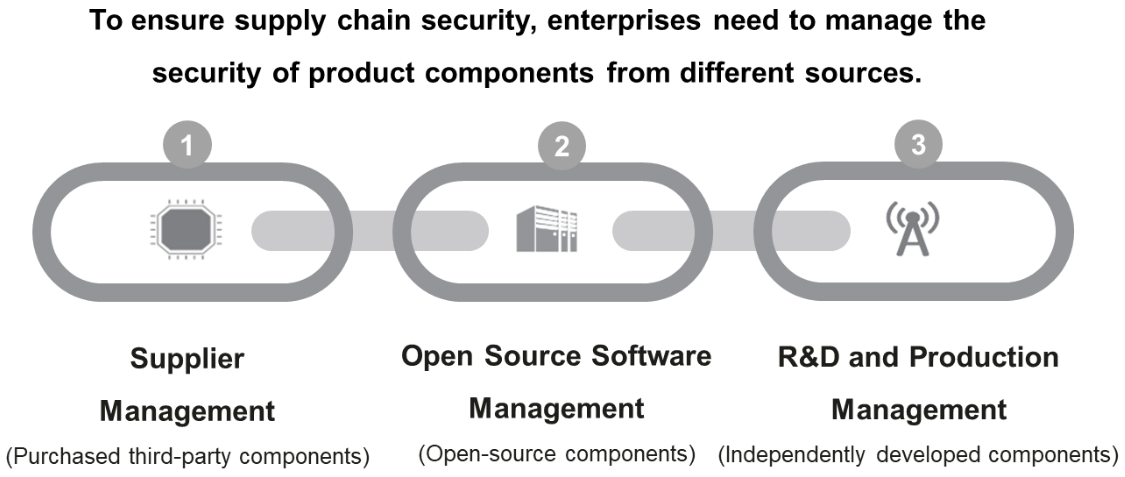
Table 2 - Key relevant criteria

No.	Country	Standard No.	Content
1	International	ISO/IEC 27036:2014	Information Technology – Security Techniques – Information Security for Supplier Relationships
2		ISO/IEC 20243	Information technology-Open Trusted Technology Provider™ Standard
3		ISO 28000	Supply Chain Security Management System Specifications
4	United States of America	NIST SP 800-161	Supply Chain Risk Management Practices for Federal Information Systems and Organizations

5		-	Software Supply Chain Security Practice Guide for Suppliers
6		-	Software Supply Chain Security Practice Guide for Developers
7		-	Customer-facing Software Supply Chain Security Practice Guide
8	China	GB/T 43698-2024	Cyber Security Technology Software Supply Chain Security Requirements
9		GB/T 43848-2024	Open Source Code Security Evaluation Method for Software Products
10		GB/T 39204-2022	Information Security Technologies Security Protection Requirements for Critical Information Infrastructure
11		GB/T 36637-2018	ICT Supply Chain Security Risk Management Guide

2 Software Supply Chain Security Governance

With the globalization of supply chain, software and hardware products become larger and more complex, and the current supply chain has become more and more complex. Every enterprise is also an important part of the supplier. Our role may be both customers and suppliers. However, no matter what phase we are in, we must do our part well. We need to manage the security of product components from different sources, including third-party component security, open source component security, and self-developed component security, to promote the security of the entire supply chain. This document provides suggestions on supply chain security governance from supplier management, open source software management, and R&D and production management.


To ensure supply chain security, enterprises need to manage the security of product components from different sources.
1
2
3
Supplier Management
(Purchased third-party components)
Open Source Software Management
(Open-source components)
R&D and Production Management
(Independently developed components)

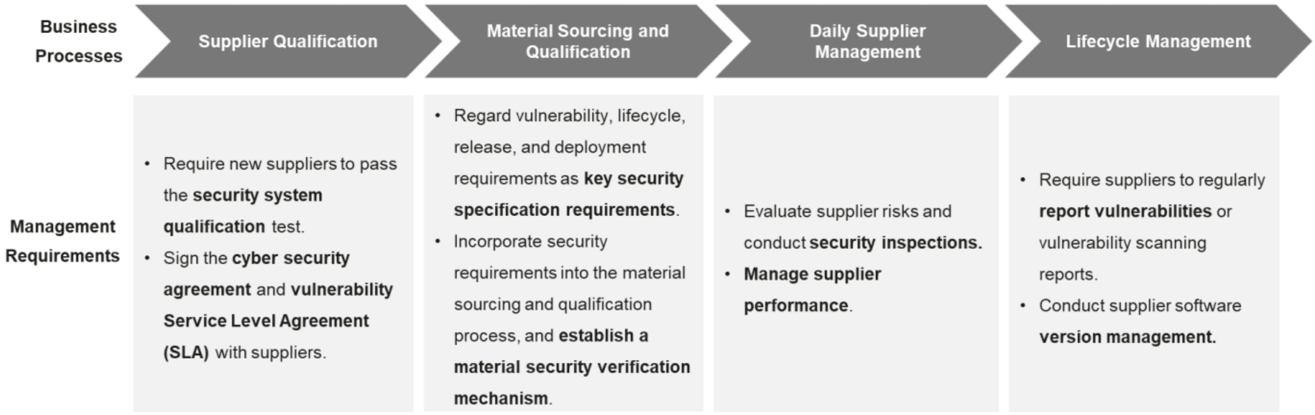
2.1 Domain 1: Supplier cyber security management

2.1.1 Overview of Supplier Cyber Security Management

Software products and services are increasingly used in key industries. The stable and secure operation of software products and services depends on the global supply chain. Globalization also brings cyber security incidents from time to time. (For example, at least 200 organizations, including Microsoft and VMware, were victims of the SolarWinds attack that was exposed at the end of 2020.) Cyber security status of materials and services of suppliers and their lower-level suppliers (e.g. viruses, vulnerabilities, poor code quality, etc.) , which determines customers' perception of cyber security capabilities of purchased products and services.

Compared with the self-developed part of software products, purchased third-party software materials are less visible, understandable, and controlled. Therefore, it is urgent to strengthen the cyber security risk management of the software supply chain.

In the E2E procurement process, including the supplier introduction phase, software material sourcing and qualification phase, daily supplier management phase, and software material lifecycle management phase, complete cyber security control requirements must be established to build strong supplier cyber security management capabilities.



Business Processes	Supplier Qualification	Material Sourcing and Qualification	Daily Supplier Management	Lifecycle Management
Management Requirements	• Require new suppliers to pass the **security system qualification** test. • Sign the **cyber security agreement** and **vulnerability Service Level Agreement (SLA)** with suppliers.	• Regard vulnerability, lifecycle, release, and deployment requirements as **key security specification requirements**. • Incorporate security requirements into the material sourcing and qualification process, and **establish a material security verification mechanism**.	• Evaluate supplier risks and conduct **security inspections.** • **Manage supplier performance**.	• Require suppliers to regularly **report vulnerabilities** or vulnerability scanning reports. • Conduct supplier software **version management.**

2.1.2 Introduce suppliers with complete security systems

2.1.2.1 The supplier has a complete security system.

The level of the supplier's cyber security system determines whether the supplier can continuously provide secure and reliable products. The cyber security system includes the following aspects:

(1) **Security system:** The supplier formulates the general cyber security policy. Establish a security organization, define security roles and responsibilities, and run the organization regularly. Pass the third-party security system certification. Establish cyber security KPIs and appraise managers at all levels.

(2) **Security development:** Suppliers establish product cyber security baselines and incorporate them into the development process.

(3) **Security test:** The supplier conducts product-specific cyber security tests and provides test reports.

(4) **Open source software security:** Suppliers establish open source software management mechanisms and provide open source software lists.

(5) **Delivery security:** Suppliers should take security control measures to prevent tampering, implanting, and package swapping during product production, including but not limited to internal software transmission, chip burning, verification, software loading, and production testing. When releasing software versions through the official website or other channels, suppliers provide integrity protection measures (such as hash verification and digital signature) to ensure software integrity and consistency.

(6) **Service security:** When providing products and services, the supplier shall comply with the local laws and regulations on protecting personal data and privacy, communication freedom, and network security, as well as the customer's security requirements, and ensure that the service process does not have security risks or problems.

(7) **Incident response:** The customer has the incident response mechanism and capability when discovering product security vulnerabilities or security incidents.

(8) **Traceability:** Suppliers clearly record and manage R&D documents, tools, source code, changes, and product versions during product requirement, design, development, test, and release to ensure traceability.

(9) **Personnel management:** Identify the list of personnel in key security positions, provide training, exams, and certification for the personnel in the list, and sign the cyber security commitment letter.

In the supplier introduction phase, the customer usually conducts an all-round audit on the supplier's cyber security system through the supplier's on-site system audit. Only the suppliers that pass the audit can be introduced as official suppliers.

2.1.2.2 Suppliers sign cyber security agreements

By signing the cyber security agreement, the supplier fulfils the legal commitment that the product complies with relevant laws, regulations, and customers' cyber security requirements. A cyber security agreement must include product security requirements, service security requirements, compensation liability, and audit clauses.

(1) Product security requirements: 1) Do not violate laws and regulations, such as illegal interception, malicious code/virus, undisclosed interfaces, and privacy infringement. 2) Basic technical requirements: secure access mechanism, data encryption, and audit management.

(2) Service security requirements: 1) Do not maliciously damage the customer network, steal customer data, and implant illegal code backdoors. 2) Authorization is required for service behaviors, such as unauthorized accounts, unauthorized operations, and unauthorized software. 3) Customer network data must comply with regulations, including illegal collection, handling, and data leakage.

(3) Compensation liability: The supplier shall fulfill all compensation liabilities for the losses caused by the supplier.

(4) Audit clauses: The supplier shall be audited by the customer or a third party entrusted by the customer.

2.1.3 Introduce secure and reliable software materials

To avoid introducing materials with security risks into their products, customers need to incorporate cyber security requirements into the material sourcing and qualification process. Vulnerability, lifecycle, release, and deployment requirements are regarded as key security specifications and incorporated into the material sourcing and qualification process. The three-layer security verification mechanism in the material sourcing and qualification phase, namely, supplier self-test, customer R&D test, and customer independent third-party test, ensures the security and trustworthiness of materials.

(1) Material security requirement specifications: 1) Service assurance requirements: vulnerability SLA requirements, lifecycle requirements, and open source list requirements; 2) Threat analysis and risk assessment: threat-based mitigation measures and fault-based mitigation measures; 3) Market access and certification requirements: market access requirements based on the host country/market and certification requirements based on the customer. 4) Network security technical requirements: legal compliance requirements, anti-illegal intrusion requirements, and security technical requirements.

(2) Compliance requirements for material testing and certification: supplier self-test, customer R&D test, and customer independent third-party test.

2.1.4 Perform routine supplier cyber security management

To ensure that suppliers can continuously provide secure products, the customer needs to perform routine management on suppliers. Common routine supplier cyber security measures in the industry include:

(1) Security performance evaluation: Score suppliers' security performance through security performance evaluation. The security performance evaluation result can be used for supplier share management and elimination management. If a cyber security incident occurs, the supplier shall compensate the customer for the impact and loss according to the cyber security agreement.

(2) Security risk level assessment: Organize the supplier security risk level assessment every year, and develop improvement measures based on the supplier security risk level assessment results and identified risks.

(3) Security system self-check: If the supplier security risk assessment result is high, or if the outstanding redline issues or serious security issues/events occur, the supplier can be required to perform additional security system self-check, or the supplier can perform cyber security system self-check before conducting cyber security system inspection. After the supplier completes the self-check, check whether the contents in the self-check form are complete. If the contents are incomplete, ask the supplier to perform the self-check again.

(4) Security system inspection: Develop the annual supplier inspection plan based on the security risk level assessment results and historical inspection results. High-risk suppliers must be included in the inspection.

(5) Improvement of system problems: Send the inspection results and suppliers' cyber security problems to the supplier security contact person, ask the supplier to perform root cause analysis, develop rectification measures, and track the problems according to the requirements for tracking the problems found in the supplier system inspection.

(6) Rectification of product redline issues: Transfer the product redline issues to suppliers, and regularly track and urge suppliers to rectify the issues. After the supplier completes the rectification of redline issues, the supplier submits the new version to the product line R&D department for testing and verification. The product line R&D personnel close the problem after the verification is passed. If the verification fails, the owner shall push the supplier to rectify the problem.

(7) Vulnerability warning: Regularly review the vulnerability feedback from suppliers and urge suppliers to proactively and promptly report security vulnerabilities to customers.

(8) Incident response: Push suppliers to establish a cyber security incident response process, ensure that suppliers can immediately initiate the incident contingency plan when cyber security vulnerabilities, incidents, and crises occur during service, notify the customer's project manager or relevant business contact persons immediately, and take effective measures to solve the problem. When upstream customers report cyber security vulnerabilities, accidents, incidents, and crises related to suppliers, urge suppliers to respond immediately, identify the impact scope of the incidents, and coordinate suppliers and customers to jointly handle the issues.

2.1.5 Manage the lifecycle of software materials

For the lifecycle management requirements of software materials, the customer requires the supplier to proactively report vulnerabilities. In addition, the supplier should respond to and fix the vulnerabilities perceived by the customer within the committed SLA time. To ensure the security and consistency of software delivery, the customer requires that the software be stored in the central warehouse through a dedicated security channel for management.

2.1.6 Introduce cyber security insurance to ensure supplier security

Supplier innovation insurance mode has gradually become a trend, not only improving the security standards of suppliers, but also providing guarantee for the supply chain security of the whole platform. This model not only effectively spreads the operational risk of suppliers, but also enhances the risk tolerance of the entire platform. According to the supplier cyber security insurance management requirements, the customer shall require suppliers to evaluate and improve the enterprise's cyber security capabilities by using the insurance method before joining the platform, increase the supplier's risk-taking capability, and do a good job in security support.

2.2 Domain 2: Open Source Software Management

2.2.1 Open Source Software Overview

Under the trend of digitalization and intelligence, the world is being defined by software. Open source software is widely used, open source software is poisoned, and vulnerabilities are exploited one after another. Open source software high-risk vulnerabilities often become network "nuclear bombs".

Take log4j as an example. The Log4j security vulnerability that broke out in late 2021 is one of the most disruptive vulnerabilities in Internet history. Since the outbreak, the impact of the Log4j vulnerability has been severe and extensive in various fields and has been increasing.

(1) Long duration: The US Cyber Security Review Committee released its first report, "Review of the Log4j Incident in December 2021", which clearly states that Log4j is an open source software that developers have integrated into millions of systems. Vulnerabilities in this ubiquitous, ubiquitous software have the power to affect companies and organizations around the world. Unfixed versions of this ubiquitous software library will remain in systems for the next decade or more. At the same time, the U.S. Cyber Security Review Board predicts that, given the prevalence of Log4j, vulnerable versions will remain in systems over the next decade, we will see the evolution of exploits, and all organizations should have the ability to discover and upgrade vulnerable software. and the ability to sustain these vulnerability management capabilities over the long term.

(2) Wide impact: According to incomplete statistics, more than 35,863 Java components of open-source software depend on Log4j, which means that more than 8% of software packages are affected by this vulnerability. The deeper the vulnerability goes in the dependency chain, the more steps there are to fix it. According to cloud security experts, there are over 1,000 attempts per second to exploit the Log4j vulnerability. The Log4j vulnerability not only affects Java-based applications and services that directly use the library, but also affects many other popular Java components and development frameworks that rely on it. As the crisis continues to ferment, the damage caused by the Log4j vulnerability cannot be accurately assessed.

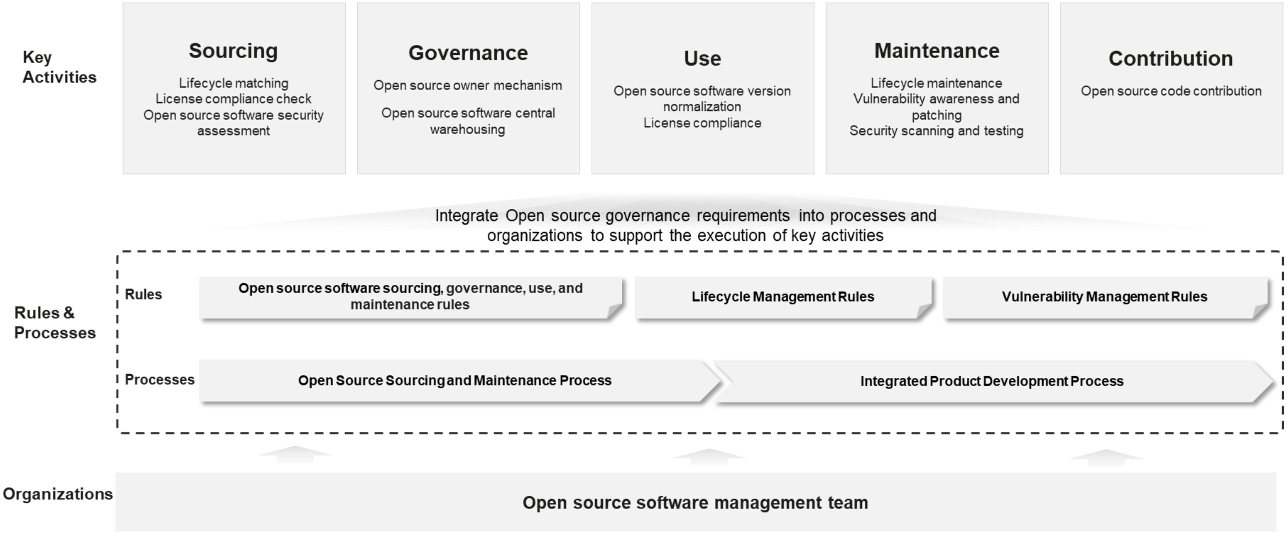
(3) High harm: Since the Log4j vulnerability broke out at the end of 2021, multiple botnet families, such as Mirai and Muhstik, have used this vulnerability to spread. At the same time, the exploit is rapidly mutating, bypassing existing mitigations and attracting more and more hackers.

In 2022, SYNOPSYS, a Black Duck software producer, reviewed 1703 code libraries from 17 industries and found that 96% of the code libraries used open source software and 84% of the code libraries contained at least one open source software vulnerability. 34% of the respondents said they had been attacked by exploiting known vulnerabilities in open source software in the past year. In code bases using open source software, 54% of the code inventory has license conflicts. (Failed to comply with the license requirements for open source software, including but not limited to segment reference, conflict between subcomponent clauses and the entire project license clauses.) 89% of code repositories contain outdated open source software. (No upgrade or fixing activities in the past four years means that the code is no longer maintained, including vulnerabilities.)

According to the preceding open source software cases and related analysis reports, the usage of open source software is increasing in various industries. Once security problems (such as high-risk vulnerabilities) occur in open source software management, the problems are fatal. Therefore, it is imperative to strengthen open source software management and build a secure and reliable open source software supply chain.

2.2.2 Building a Secure and Trusted Open Source Software Supply Chain

Open source software management: Build a secure and reliable open source software supply chain through open source sourcing, use, maintenance, and community feedback. In the sourcing phase, select mainstream, active, and high-quality open source components. During the use and maintenance process, manage the lifecycle of open source components from the full product perspective, fulfil obligations by complying with open source licenses, and continuously detect and handle security vulnerabilities to ensure the security and compliance of open source use. In addition, we need to build these engineering capabilities into processes, tools, and organizations to build a secure and reliable open source software supply chain.


Key Activities
Sourcing
Lifecycle matching
License compliance check
Open source software security assessment
Governance
Open source owner mechanism
Open source software central warehousing
Use
Open source software version normalization
License compliance
Maintenance
Lifecycle maintenance
Vulnerability awareness and patching
Security scanning and testing
Contribution
Open source code contribution
Integrate Open source governance requirements into processes and organizations to support the execution of key activities
Rules & Processes
Rules
Open source software sourcing, governance, use, and maintenance rules
Lifecycle Management Rules
Vulnerability Management Rules
Processes
Open Source Sourcing and Maintenance Process
Integrated Product Development Process
Organizations
Open source software management team

2.2.2.1 Open Source Software Selection Principles

(1) Matching between product objectives and open source software. According to industry practice, if more than 20% lines of code need to be modified in open source software, it is better to purchase "commercial software" that better matches the functions. Avoid additional costs incurred during future software maintenance.

(2) Check whether the license of open source software can be complied with. For example, if the cloud service uses open source software that complies with the SSPL protocol, consider whether the code can be opened to the public.

(3) Check whether the quality and security of open source software meet product requirements.

(4) Check whether the open source community is active and whether the software version can be maintained with the help of the community to avoid introducing software that has died out (no one is operated by the open source community) or versions that are no longer maintained by the open source community.

2.2.2.2 Open source software selection and governance

(1) Technical Ecosystem Assessment of Open Source Software

1) Check whether the technical architecture and ideas are advanced in the industry and comply with the existing technical planning of the product.

2) Check whether the open source software is used by commercial users of a certain scale and whether there are mature application scenarios and cases.

(2) Compliance Assessment of Open Source Software

1) Check whether the source for obtaining open source software is valid and whether the source can be obtained continuously.

2) Check whether a specific license is available. Do not introduce open source software without a license. In commercial use scenarios, whether the product can comply with the license and fulfil corresponding obligations, including but not limited to the use declaration and code open source obligations.

(3) Life Cycle Assessment of Open Source Software

1) Check whether the open source software community is maintained by developers and whether the open source software community has continuous version/code updates. Check whether the lifecycle of open source software meets product requirements.

2) The LTS version of the community in the dimension is preferred. For communities without LTS versions, select the latest stable version as possible.

(4) Open source software security evaluation

1) Scanning for viruses and malware

2) Check whether the open source community has channels for seeking and resolving defects/security issues.

3) Under the same conditions, select a version that has no or few known vulnerabilities.

4) Check whether the product has corresponding solutions and policies for handling known vulnerabilities.

(5) Open source software entity asset and metadata governance:

1) Check whether the source of the introduced open source software (source code package or binary package) is reliable (from the official release channel of the community). Check

whether the introduced open source software is consistent with the community (through the HASH check). Check whether the entity is backed up in a dedicated software repository as the only source used by the product.

2) Obtain and record open source software metadata information from the community as the basic input for subsequent management, including but not limited to open source software name, version, official website, code hosting address, binary package download address, license information, copyright information, and dependency information.

2.2.2.3 Use of open source software

2.2.2.3.1 Currently, enterprises use open source in the following two modes:

(1) Development based on open source software

Many enterprises develop open source software and build their own commercial products on the basis of compliance. For example, most cloud vendors build container-related cloud products based on the open source project Kubernetes under the CNCF Foundation. In addition, they have developed many open source or closed source plug-ins and operators to meet different customer requirements. As a result, the same type of products of these cloud vendors are different. Competitiveness differences are formed based on different customers and scenarios. However, when using Kubernetes, you need to continuously follow up on Kubernetes version changes, which consumes a lot of energy. In this case, cloud vendors open source their own features to the upstream to avoid forming different maintenance branches with the upstream community.

(2) Invoking open source software functions

Some enterprise-developed code invokes its functionality at run time (some custom modifications may be made). For example, when Nextjs is used to develop page applications, a large number of JavaScript libraries are integrated. Some programs developed in Linux invoke the MD5 library of OpenSSL to calculate the hash value for verification.

In any scenario, open source software must be used in a standardized manner to control compliance and security risks and reduce use costs.

2.2.2.3.2 Standardized use of open source software

(1) Analyze all the files involved in the product build (can be scanned by the open source component analysis tool) and record the open source software component information used in the product software version as the basic input for subsequent open source software management.

(2) Open source code is isolated from self-developed code and stored in an independent directory. Do not add open source code fragments to self-developed code.
Open source code fragments are mixed into self-developed code, which is difficult to trace and trace. In terms of compliance and security, the following risks may be brought:

a) **License compliance:** Open source code fragments are also subject to open source licenses. Using fragments may cause the corresponding license compliance to be ignored.

b) **Security traceability:** Open source code fragments may also have vulnerabilities, and the code used by the fragments is difficult to trace and fix.

(3) All modifications to open source software are recorded. The native code of open source software and the modified product code are clearly distinguished, facilitating the identification and control of the modified content. A community participation channel should be established for the product team, and the modified code can be fed back to the open source community to minimize the number of proprietary modifications.

(4) As a part of the product, open source software is scanned and tested to identify and handle security issues.

(5) Based on product application scenarios, comply with the open source software license and fulfil the obligations specified by the license, including but not limited to the use declaration and code open source obligations.

2.2.2.4 Open source software maintenance

To use open source software, products must have open source maintenance capabilities. Resources must be invested in the product lifecycle. Open source software vulnerabilities must be continuously detected through the community official website and industry vulnerability library, and the vulnerabilities must be fixed in a timely manner.

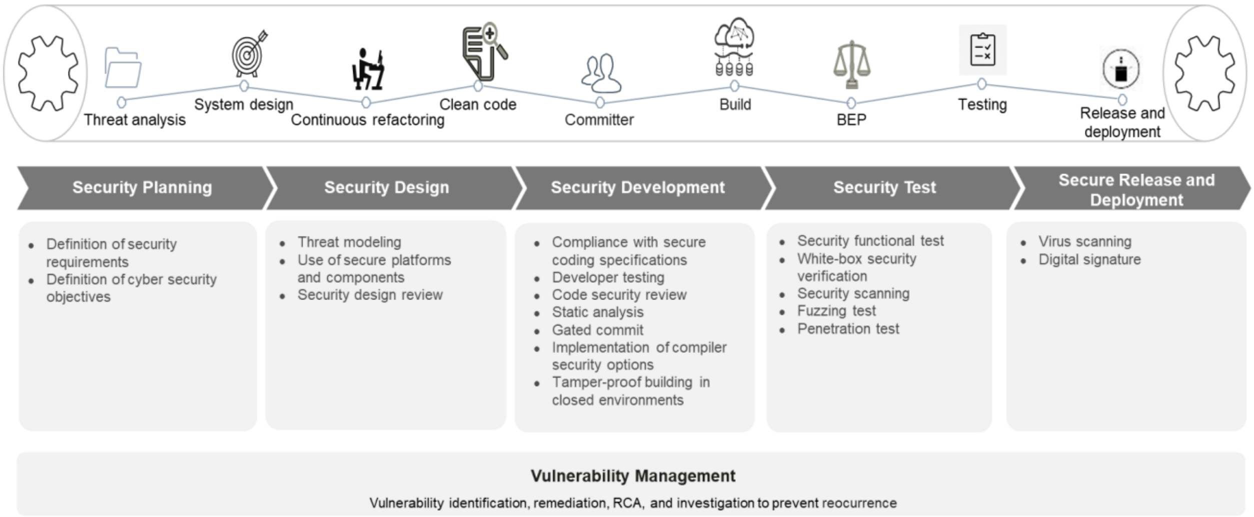
For the vulnerabilities disclosed and resolved by the community, the product team shall promptly synchronize the patch or version to fix them. For the vulnerabilities disclosed by the community but not provided with a fixing solution, the product team can fix or mitigate the vulnerabilities based on the risks. Vulnerability response/fixing time meets the SLA requirements agreed by the product and customer or general industry requirements.

2.3 Domain 3: R&D and Production Management

2.3.1 Adopt the product security development process to build secure and reliable high-quality products

With the acceleration of all-cloud, digitalization, and software-defined everything, cyber attacks and threats are becoming the norm. Trustworthiness and network resilience will become more and more important. Compared with new functions and features, secure and reliable products will become the prerequisite for customers to purchase in the future.

To build secure and reliable high-quality products, we must rely on the end-to-end product security development process. The security development process includes security planning, security design, security development, security testing, and security release and deployment.


Threat analysis
System design
Continuous refactoring
Clean code
Committer
Build
BEP
Testing
Release and deployment
Security Planning
Security Design
Security Development
Security Test
Secure Release and Deployment
• Definition of security requirements
• Definition of cyber security objectives
• Threat modeling
• Use of secure platforms and components
• Security design review
• Compliance with secure coding specifications
• Developer testing
• Code security review
• Static analysis
• Gated commit
• Implementation of compiler security options
• Tamper-proof building in closed environments
• Security functional test
• White-box security verification
• Security scanning
• Fuzzing test
• Penetration test
• Virus scanning
• Digital signature
Vulnerability Management
Vulnerability identification, remediation, RCA, and investigation to prevent reocurrence

(1) **Security planning:** In the planning phase, the trustworthiness definition of the product or system must be completed, and the security requirements and functions that the system must meet must be specified and incorporated into the requirement document.

(2) **Security design:** In the design phase, threat modelling and analysis are performed, security risks are identified, and system security architecture and security policies are formulated, including access control, identity authentication, and data encryption.

(3) **Security development:** In the coding phase, code must be compiled in compliance with secure coding specifications, static code scanning, and secure compilation selection must be enabled to enhance code quality and avoid common security vulnerabilities.

(4) **Security testing:** In the testing phase, security testing, including black-box testing and white-box testing, needs to be performed based on threat modelling analysis to detect and fix security vulnerabilities.

(5) **Security release and deployment:** During the release and deployment phase, security configuration hardening, including network configuration and access control configuration, needs to be performed to ensure system security.

2.3.2 Trustworthy construction

While building is not usually the core part of enterprise software engineering, modern software often introduces open source software and enterprises have a lot of code inherited from previous versions, tracing all of that code and managing the compilation environment for that code is a particularly complex challenge. Therefore, enterprises often attach importance to them as important problems of software engineering and solve them.

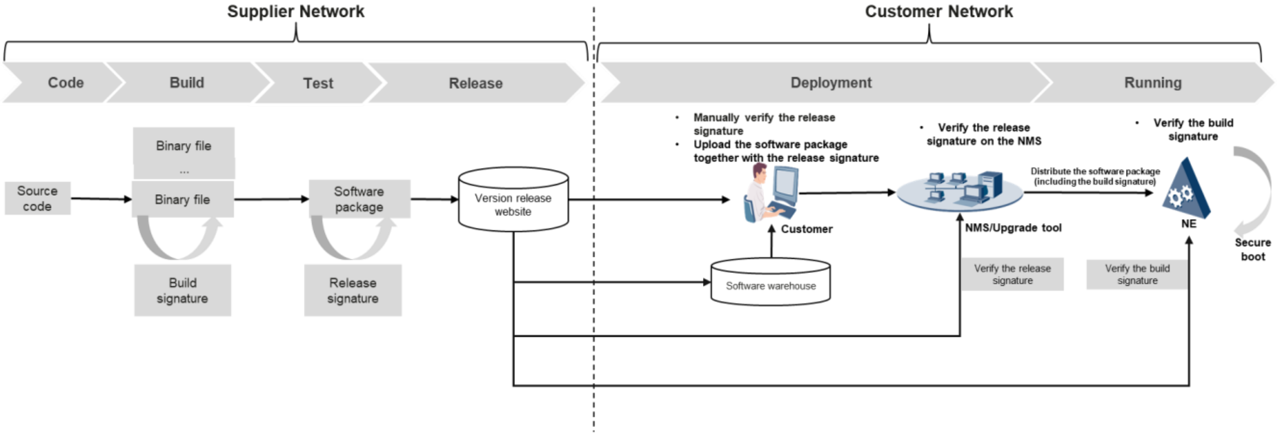
By centralizing the available open source software and product development source code into the unified repository of the company for management, using scripts (as part of the code officially managed by the version) to automatically generate the build environment, and placing the build environment in a fully protected Working in an environment isolated from the office network, the product can initiate a complete build activity by using the platform code and product code through the unique compilation entry file to generate all software packages of a single product used by the customer. In this way, the product software generation process is completely repeatable and all open source components and self-developed code are completely traceable.

The customer has also conducted unique research in determining code and binary consistency, which effectively compares two packages generated using the fully automated build process in two independently generated isolated build environments, and identifies any suspicious differences. Therefore, the risk of tampering with the software package during the automatic build process can be effectively avoided.

Improvements in all these build domains benefit from the improvement of customers' build engineering capabilities, including promoting the unified build framework and build project directory structure, complying with the unified build specifications of Huawei, and automatically monitoring through the check tool.

2.3.3 Software Integrity Protection

Implement end-to-end anti-tampering/implantation/malware/spoofing through external source management, internal R&D activities, delivery, and deployment integrity protection, and mitigate software supply chain security risks. In addition, integrity protection needs to be extended to customers. Customers can cooperate with each other in software package direct access, signature verification, and digital certificate management.


Supplier Network
Customer Network
Code
Build
Test
Release
Deployment
Running
Binary file
...
Source code
Binary file
Build signature
Software package
Release signature
Version release website
Manually verify the release signature
Upload the software package together with the release signature
Customer
Software warehouse
Verify the release signature on the NMS
NMS/Upgrade tool
Verify the release signature
Distribute the software package (including the build signature)
Verify the build signature
NE
Verify the build signature
Secure boot

External source: All open-source software and third-party software used in product compilation and building come from the supplier's official website or legal hosting website. Malware scanning and signature protection are performed before product introduction and stored in the central warehouse.

Internal R&D activities: All code submitted has been reviewed and approved, meeting SOD requirements. 100% product build components, including self-developed code, build tools, open source software, and third-party software, are downloaded from the central warehouse. The build process is automated, the build environment is closed, and the build results pass the BEP consistency check.

Delivery and deployment: All released software packages have digital signatures. The eSight and deployment tool automatically verify the digital signature when the software package is imported. The digital signature is also verified during the software upgrade.

2.3.4 Vulnerability management

2.3.4.1 Enterprises need to manage the upstream, do themselves well, and serve the downstream well.

Enterprises can do a good job in the whole supply chain loophole management from three aspects: managing upstream, doing a good job in themselves and serving downstream.

First of all, managing upstream systems is to ensure the security and trustworthiness of open source and third-party components used by enterprises. Vulnerability management activities are conducted for third-party and open source components in the four phases of selection, use, cultivation, and retention.

For example, when selecting third-party components, sign the Trustworthiness Agreement with the supplier to require the supplier to meet the customer's vulnerability notification and response requirements. In addition, qualify the supplier's security capabilities and use the qualification results as the basis for supplier selection.

When third-party and open-source components are used, ensure that supplier software and patches are allowed to enter the customer software repository only after strict security scanning is performed. In addition, irregular vulnerability scanning is performed to check the supplier's vulnerability management capability.

Establish a supplier performance management mechanism and incorporate the vulnerability response capability of suppliers into their long-term performance management. For suppliers that do not meet customers' vulnerability management requirements, a corresponding elimination mechanism is required.

Second, you have to be good at yourself.

To do well yourself is to do well in the following two aspects:

(1) Develop secure products: Combine forward security design and reverse security testing to ensure that the products are free of known vulnerabilities (or risks are controllable) before release.

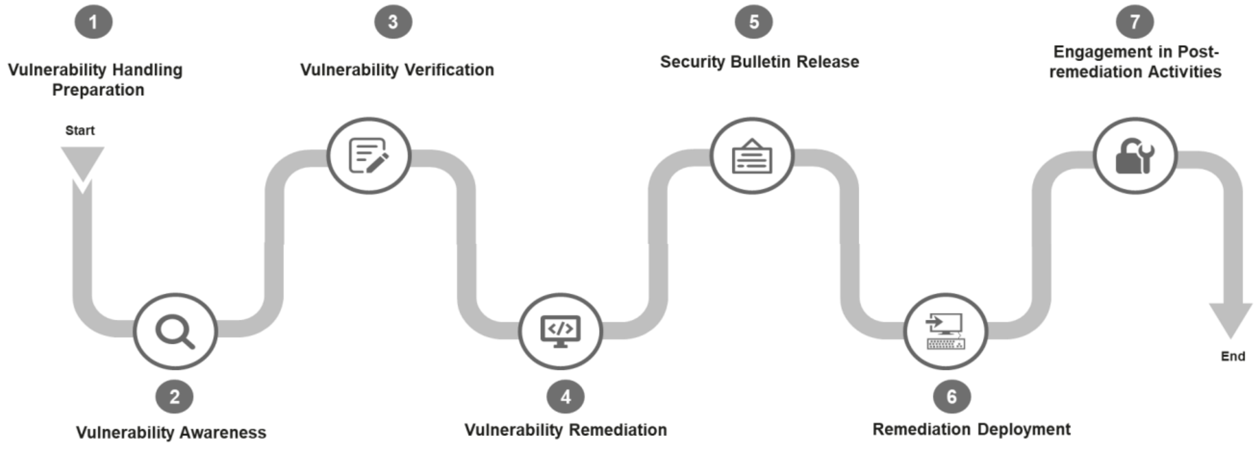
(2) Vulnerabilities are continuously perceived, verified, patched, and disclosed in the lifecycle phase: After the product is released (in the lifecycle phase), known vulnerabilities are continuously perceived, affected, and patched in a timely manner and released the patching solution to customers.

Finally, as a part of the supply chain, we serve downstream customers well, help customers improve security awareness, enhance vulnerability management capabilities, continuously provide vulnerability mitigation solutions for customers throughout the product lifecycle, and provide technical support for risk mitigation on customers' live networks.

By managing the upstream, serving the downstream well, and forming closed-loop management of vulnerability management in the entire supply chain.

2.3.4.2 Vulnerability management process: Seven-step vulnerability management, in compliance with ISO/IEC 29147/30111.

It is recommended that the vulnerability management process comply with industry standards and best practices. Specifically, the following seven-step vulnerability management process fully complies with ISO/IEC 29147/30111.


1
Vulnerability Handling Preparation
Start
3
Vulnerability Verification
5
Security Bulletin Release
7
Engagement in Post-remediation Activities
End
2
Vulnerability Awareness
4
Vulnerability Remediation
6
Remediation Deployment

Establish corporate-level management policies and organizations and develop relevant capabilities during the preparation phase of vulnerability handling.

Vulnerabilities can be quickly detected through diversified vulnerability awareness capabilities, including vulnerability awareness through public channels and active vulnerability reward program activities.

In the vulnerability verification phase, evaluate the vulnerability severity based on the CVSS standard, determine the affected products and versions, and create trouble tickets for tracing.

The product version team provides workarounds and patch solutions for the product version based on the vulnerability trouble ticket until the trouble ticket is closed.

Based on the need-to-know and impact notification principles, disclose vulnerability information to affected customers in a timely manner through multiple methods, such as SA, SN, and RN.

Provide necessary technical support to help customers fix vulnerabilities on the live network during the rectification deployment phase.

After the vulnerability is fixed, continuously track the vulnerability fixing status and risk mitigation status, analyze the root cause of the vulnerability, and incorporate mitigation measures into the security development lifecycle activities to prevent the recurrence of similar vulnerabilities.

2.3.4.3 Support customers' cyber security risk mitigation through upstream and downstream collaboration based on the Need-to-Know vulnerability disclosure principle.

Vulnerability handling requires upstream and downstream collaboration. Vulnerability disclosure is a key part of vulnerability collaboration. Vulnerability disclosure is not only the guarantee and practice of customers' right to know, but also the trigger input for customers' live network rectification.

Vulnerability disclosure is recommended to follow the three principles: impact notification, harm reduction and accident avoidance, and transparent disclosure.

Adopt disclosure policies based on vulnerability priorities. For common vulnerabilities, regularly disclose the vulnerabilities through the disclosure website. For high-risk vulnerabilities, adopt a more proactive communication strategy. Frontline personnel proactively communicate with stakeholders authorized by the customer.

In addition to providing sufficient vulnerability information for customers to make risk decisions based on their own asset environment, the supplier's technical support team will also support the customer's O&M team to mitigate live network risks based on the risk mitigation or fixing deployment solution determined by the customer.

3 Prospects for the future

With the rapid development of digitalization and intelligence, software systems are becoming more and more complex, and hardware integration is becoming more and more high. We need to pay attention not only to the attack risks of the software supply chain, but also to the physical security of hardware. Anti-tampering and anti-implanting have become a systematic security issue. By analyzing the security incidents of software supply chain in recent years, we can find that software supply chain attacks have the characteristics of various attack modes, wide attack areas, large impact scope, strong concealment, unpredictable attack, and even new risks of physical attacks by software controlling hardware. The following uses artificial intelligence (AI) and the Internet of Things (IoT) as examples to describe the new risks brought by the development of new technologies to supply chain security.

With the rapid development of artificial intelligence, especially the breakthrough of generative artificial intelligence, its complex effects are also emerging, not only releasing huge technical dividends, but also resulting in security issues beyond the previous scope, and the process of artificial intelligence trustworthiness and security governance is also accelerating. LLM provides a new path for system attacks. Hackers may attack supply chain systems by exploiting traditional vulnerabilities and big model manipulation. At the same time, low-level hackers can use AI to carry out difficult supply chain attacks. For large models, the trust boundary of the LLM system needs to be sorted out and analyzed in detail the external sources of data and the data transfer in the system, especially the part that can be used as the input of the model, for protection and control. Pay special attention to the security of components (such as RAG, plug-in, and production platform) involved in large models. Minimize the permission on each component. Use sandboxes to isolate components if necessary. For large model testing, you need to pay attention to model deserialization and unauthorized data. You can also use tools to poison models during the use phase.

The Internet of Things (IoT) realizes the convergence of virtual and physical worlds, and is an important link between terminal-oriented sensor networks and data-oriented applications driven by Internet technology. In modern society, the introduction of IoT should give priority to the interests of IoT stakeholders and meet their security and privacy requirements. The latter is critical in the Internet of Things, which defines the security boundaries of open systems that combine public, private, and community. IoT terminals usually have limited resources and are vulnerable to malicious attacks. An attacker can intrude into an IoT terminal and launch attacks on other IoT terminals. As a result, intrusions to a single IoT terminal can gradually spread to thousands of IoT terminals like an infectious disease. Attackers can exploit a large network of infected IoT endpoints to launch attacks on services or platforms used or relied on by any device. IoT security comes from the technical challenges at the endpoint, network, and platform levels, as well as the process challenges of integrating security technologies in an end-to-end manner. To face these challenges, the following three key security technologies and one process capability are critical: (1) Terminal perspective: moderate terminal defense capability; (2) Network perspective: malicious terminal detection and isolation; (3) Platform perspective: platform and data protection; (4) Process perspective: From the perspective of IoT security, security operations and management should address the challenges at all the above levels and support customers to obtain the security assurance they need.

Appendix A Abbreviations

Abbreviations	Full name
AI	Artificial Intelligence
AIGC	Artificial Intelligence Generated Content
APT	Advanced Persistent Threat
BEP	Build Equivalence Project
BSIMM	Building Security In Maturity Model
CNCF	Cloud Native Computing Foundation
CVSS	Common Vulnerability Scoring System
GDPR	General Data Protection Regulation
ICT	Information and Communications Technology
IEC	International Electrotechnical Commission
IoT	Internet of Things
ISO	International Standardization Organization
KPI	Key Performance Indicator
LLM	Large Language Model
LTS	Long Term Support
NIST	National Institute of Standards and Technology
OWASP	Open Web Application Security Project
RAG	Retrieval Augmented Generation
RN	Release Notes
SA	Security Advisory
SN	Security Notice
SAMM	Software Assurance Maturity Model
SLA	Service Level Agreement
SSDF	Secure Software Development Framework
SSPL	Server Side Public License

Appendix B Glossary Expansion

Abbreviations	Definitions
RAG	Retrieval Augmented Generation (RAG) is a technology that combines information retrieval and model generation. It provides contextual information for model generation by retrieving relevant documents from information retrieval system, thus improving the quality of model generation.
SSPL	The Server Side Public License (SSPL) is a source-available software license introduced by MongoDB Inc. in 2018. It is a modified version of the GNU General Public License version 3 (GPL v3), which mandates

	that any third-party "service" that incorporates SSPL-licensed software must release the entirety of their source code under the SSPL.

Appendix C Bibliography

The OIC CERT supply chain security framework mainly refers to the following standards:

(1) ISO/IEC 27001 Information Security Management System

(2) ISO/IEC 27701 Privacy Information Management

(3) ISO 28000 Supply Chain Security Management System

(4) ISO/IEC 29147 Vulnerability Disclosure

(5) ISO/IEC 30111 Vulnerability Handling Process

(6) GB/T 43698-2024 Network security technology-Software supply chain security requirements

(7) GB/T 43848-2024 Network security technology--Open source code security evaluation method for software products

© OIC-CERT Supply Chain Security Framework 2024 All Rights Reserved