Guidelines on the Roles Expected of Cyber Infrastructure Providers

— Appropriate division of roles and responsibilities between customers and cyber infrastructure providers to ensure cybersecurity and improve resilience in software development, supply, and operation —

March. 2026

Industrial Cybersecurity WG1, Ministry of Economy, Trade and Industry/Joint Working Group,
National Cybersecurity Office
Study Group on the Roles Required of
Cyber Infrastructure Providers

Table of Contents

1. Preamble

1.1. Background and objective

(1) Need to improve cybersecurity[1] resilience

In recent years, cyberattacks have become more intense and diverse, with recent attacks targeting the core software that digitally supports all social activities and the potential vulnerabilities in supply chains. Owing to the strong dependence on software for the management of various information and telecommunications systems and services, cyberattacks can undermine the reliability of such digital platforms and severely affect people's lives and economic activities, as well as critical infrastructure. Table 1 lists typical examples.

Table 1 Typical examples of cyberattacks

Example	Summary
Apache Log4J vulnerability	Apache Log4J is a logging library used worldwide. In 2021, a serious vulnerability that allowed attackers to execute arbitrary code remotely was discovered within the library and exploited. It was incorporated and used in various types of software in multi-layered software supply chains. This incident highlights the need to discover, track, and fix vulnerabilities.
Tampering with a software update from Software Vendor A	A legitimate software update was tampered with by intruding into the software vendor system, affecting all organizational functions using the software. The incident suggests the importance of ensuring the security of development and operation environments in the software supply chain.
Encryption and leakage of patient information held by Hospital B	An in-hospital network was infiltrated via a VPN device unpatched against known vulnerabilities, causing disruptions in medical treatment. This case illustrates the importance of proactive management of software security by the hospital (customer) and the importance of information provision by the provider.

[1] Cybersecurity refers to measures taken to prevent leakage, loss, and damage of information through electromagnetic means and to ensure the safety and reliability of systems and networks that handle such information, as well as the maintenance and management of such measures. Refer to Article 2 of the Basic Act on Cybersecurity.

These examples show that cyberattacks on software can target various areas: the development phase, which includes the design of systems and services; the construction, maintenance, and operation phases; supply chains between users, software developers, and suppliers; and the contract phase. Thus, it is difficult to develop countermeasures covering all phases. To appropriately respond to such difficulty, followings are required: public-private collaboration on cyber infrastructure provider’s role and ideal, risk management on cyber security risks, and balancing it with costs. In this regard, the United States (US) has recently created standards and guidelines to strengthen security in software development and supply chains. In addition, the European Union (EU) is accelerating the development of institutional systems in connection with the reinforcement of cybersecurity measures for digital products and services; for instance, the Cyber Resilience Act (CRA) is scheduled to come into force in 2024 and will be fully implemented by 2027. Furthermore, with the concept of "secure by design" gaining international support[2], we are entering an era in which companies are expected to not only protect themselves from cyberattacks but also take cybersecurity measures for the software products and services that they offer.

Article 7 of Japan's Basic Act on Cybersecurity states the responsibilities of cyber-related businesses and suppliers of information systems[3], and in particular, that suppliers of information systems are obligated to make efforts to provide cybersecurity assurances to users of information systems; however, there is no document specifying the roles of businesses (hereinafter referred to as "cyber infrastructure providers[4]") that provide certain social infrastructure functions through the development, supply, and operation of software, including software design, in providing cybersecurity measures for software products and services at each phase of development, supply, and operation.

The purpose of the Guidelines is to improve the resilience and fundamental cybersecurity assurances of cyber infrastructure providers by organizing and explaining the roles expected of them.

[2] In October 2023, government agencies and other entities from 13 countries, including Japan and the US, co-signed guidance that summarizes recommendations for ensuring the security of IT products (especially software) from the design stage. The guidance is available on the US CISA website (https://www.cisa.gov/securebydesign).
[3] Cyber-related businesses are entities that conduct business related to the development of the Internet and other advanced information and communication networks, utilization of information and communication technology, or cybersecurity. Suppliers of information systems are suppliers of information systems, or computers or programs that constitute part of an information system, information and communication networks, or electromagnetic storage media.
[4] Cyber infrastructure providers are businesses that develop and provide information and communication systems, software products, and ICT services that are widely used in society, including government agencies and critical infrastructure operators, as well as operators involved in the life cycle and supply chain of the software for such information and communication systems, among the operators whose responsibilities are stipulated for cyber-related businesses, etc. in the Basic Act on Cybersecurity (those who conduct development of the Internet and other advanced information and communication networks, utilization of information and communication technology, or business related to cybersecurity).

1.2. Positioning of the Guidelines

(1) Development system

The Guidelines state the responsibilities (items similar to basic principles) of cyber infrastructure providers and their supply chains that provide IT/OT systems, software products, or ICT services to customers (including government agencies and critical infrastructure operators[5]). So that cyber infrastructure providers may able to provide an appropriate division of roles between operators and customers to promote effective cybersecurity measures intended for safeguarding the software in supply chains. In addition, the guidelines outline systematic measures that are essential for risk management, which involves identifying and assessing cybersecurity-related risks and implementing appropriate risk responses to reduce residual risks to within tolerable levels. Based on the responsibilities expected of the operators and customers and the requirements[6] for fulfilling their responsibilities, it is expected that operators and customers will recognize their respective roles, share accurate information, and work together to ensure security, which will lead to improved capabilities in responding to cyberattacks.

In other countries, security measures for software supply chains are implemented not only through technical initiatives but also through the direct imposition of discipline on companies. However, as there are currently no laws in Japan that directly regulate cyber-related businesses involved in software supply chains, the Guidelines will provide a reference for businesses, companies, and related parties to ensure the effectiveness of cybersecurity measures.[7]

(2) How to use

The Guidelines are intended for use by cyber infrastructure providers and customers. At the time of use, the division of roles in the development, provision, and operation of the software through its life cycle is determined based on the characteristics of the intended software and contractual form for using the software. Cyber infrastructure providers shall seek to reach agreements with customers as necessary and understand the scope of the responsibilities that they are required to fulfill.

[5] Critical infrastructure operators refer to the critical social infrastructure operators defined in Article 3, Paragraph 1 of the Basic Act on Cybersecurity. They form the foundation of people's lives and economic activity, and conduct businesses related to objects that are likely to have a significant impact on people's lives or economic activity if their functions are suspended or impaired.

[6] The requirements described in these guidelines provide examples of effective initiatives for fulfilling responsibilities in software development, supply and operation. They are not intended to impose certain obligations, sanctions, burdens, or other disadvantages.

[7] Article 7, Paragraph 1 of the Basic Act on Cybersecurity (Responsibilities of Business Entities Related to Cyberspace and Other Business Entities) states, "In line with the basic principles, business entities related to cyberspace and other business entities are to endeavor independently and actively to ensure cybersecurity, while also endeavoring to cooperate with the cybersecurity policy that the national government or local governments implement, in the course of their business activities" and Paragraph 2 (Responsibilities of Information System Suppliers) states, "Suppliers of the information systems, or computers or programs that constitute a part of those systems, information and communications networks, or electronic or magnetic recording medium (referred to below as "information systems, etc." in this paragraph) are to endeavor, in order to prevent damage to the information systems, etc. that they have supplied due to threats to cybersecurity, to support the measures taken by the users of information systems, etc. to ensure cybersecurity, including designing and developing information systems, etc. that take into consideration the measures taken by the users to ensure their security and reliability, and continuously providing the necessary information for their proper maintenance and management." These guidelines reflect the study group's understanding of these provisions, and aim to organize and explain the roles of businesses, companies and related parties to strengthen their resilience.

- Cyber infrastructure providers

  Cyber infrastructure providers can use the Guidelines as a tool to enhance the security measures in the software supply chain. The requirements listed in the Guidelines can be used to confirm whether the efforts of their own organization and those in the software supply chain are sufficient.

  To advance this initiative, it is necessary that the cyber infrastructure providers establish secure software development and maintenance processes throughout the entire supply chain (including software component suppliers, software development contractors, and development outsourcing partners); in addition, their own organization and those in the software supply chain will need to make appropriate investments in process changes, such as changes in software development regulations. It is necessary to implement these initiatives while considering the medium- to long-term investment effects.

  By reducing software vulnerabilities through these approaches, it is possible to minimize the costs involved in creating patches to fix software vulnerabilities in the short term and software maintenance in the long term. Furthermore, when customers use security-conscious software, which reduces configuration risks and other operational errors, customer security will improve and thereby increase the trust in cyber infrastructure providers.

- Customers

  Customers are expected to use these guidelines, particularly at the procurement stage of software products and services. Customers can use the Guidelines to select appropriate software developers by either following the requirements listed in the Guidelines as specifications when developing, supplying, and operating software within their own organization, or using the requirements specified in the Guidelines as a check list to evaluate the efforts of cyber infrastructure providers from whom software and services are procured. Moreover, customers will be able to manage cybersecurity risks and reduce the operational burden of implementing vulnerability fixing patches and other measures by selecting appropriate operators through these initiatives.

  In addition, if the customer is an organization with its own software development, supply, and operation departments, they can address cybersecurity risks throughout the entire software life cycle by independently carrying out activities based on the responsibilities and roles specified in the Guidelines for cyber infrastructure providers and customers.

  It is important to note that the cost of risk management implemented by cyber infrastructure providers also includes the compensation provided to other related operators in software supply chains for incorporating security measures. In addition, customers must make appropriate investments, such as managing their own risks and developing secure procurement and operational processes and resources. It is important even for customers to recognize the importance of ensuring software security in conducting business and to take the stance of strengthening security while appropriately controlling the expansion of risk response costs by paying particular attention to efforts related to the requirements of the Guidelines .

1.3. Applicable objects

(1) Scope of software

In the Guidelines, targeted software are those developed and maintained over the software life cycle (Table 2), such as software products, software services, firmware embedded in IT/OT/IoT devices, and the software that constitutes an IT/OT system or ICT service. (Hereinafter, the terms "systems," "services," and "system services" are used to refer to IT/OT systems and ICT systems, respectively, or collectively.)

Table 2 Classification of target software

Name	Description
Software product	Software provided to customers as a product
Software service	IT services used directly by customers, such as a cloud service
Embedded software	Embedded software and firmware provided as part of a hardware product[8] such as an IT/OT/IoT device
Software that constitutes a system or service	Software that constitutes an IT/OT system or ICT service. Application software developed specifically for applications such as web programs[9] or software such as an operating system, software package, software library, and open-source software that is built by a developer and integrated into a system and provided as a system component

(2) Target businesses

The Guidelines assume that "cyber infrastructure providers" involved in the development, supply, and operation of software, including its design, are intended targets. To improve the software cybersecurity resilience, cyber infrastructure providers are required to strengthen relationships in various aspects, not only in terms of involvement aimed at protection against incidents, but also as collaborators in information collection, analysis, and response coordination in pre- and post-incident responses. In the Guidelines, cyber infrastructure providers are classified into three main roles: developer, supplier, and operator. To promote effective cybersecurity measures in a software supply chain, there must be an appropriate division of roles between cyber infrastructure providers and customers, as well as cooperation among other related organizations such as industry partners of cyber infrastructure providers; therefore, other stakeholders are also considered as targets. Table 3 shows the classification of stakeholders.

[8] This includes various types of connected devices (such as network, IoT, control, testing, transport, medical, and other connected devices). The "Common Standards for Cybersecurity Measures for Government Agencies and Related Agencies" define the "hardware" to be procured as "server equipment, terminals, communication line equipment, multifunction printers, equipment for specific purposes, software, etc." and calls for security measures for software that manages or controls these types of information system infrastructure.
[9] In web design work, programming with scripting languages, etc. may be carried out. In such a case, responsibilities equivalent to those of a developer are required.

Table 3 Classification of cyber infrastructure providers and stakeholders

Classification	Name	Description
Cyber infrastructure providers	Developer	A business or personnel engaged in designing, development, or integration of software products, software services, embedded software, and/or systems and services that are composed of such software. Developers are entities that develop or integrate software for a software development vendor, software service provider, device development vendor, software and system development contractor, software component developer, infrastructure operator, development department for in-house developed software, etc.
	Supplier	A business or personnel[10] that provides customers with software products, software services, embedded software (including hardware products), or systems and services that are composed of such software. Suppliers are entities that provide software or systems/services to a sales company of software products and devices; they include software/software service providers, system development and operation contractors, infrastructure operators, and software development vendors.
	Operator	A business or personnel that performs tasks to support the operation of systems and services for customers[11].
Stakeholders	Customers	Businesses who are the main entities of software utilization, like government agencies, critical infrastructure operators.
	Other related organizations	Organizations responsible for supporting the improvement of cyber resilience.

(3) Typical division of roles in a system

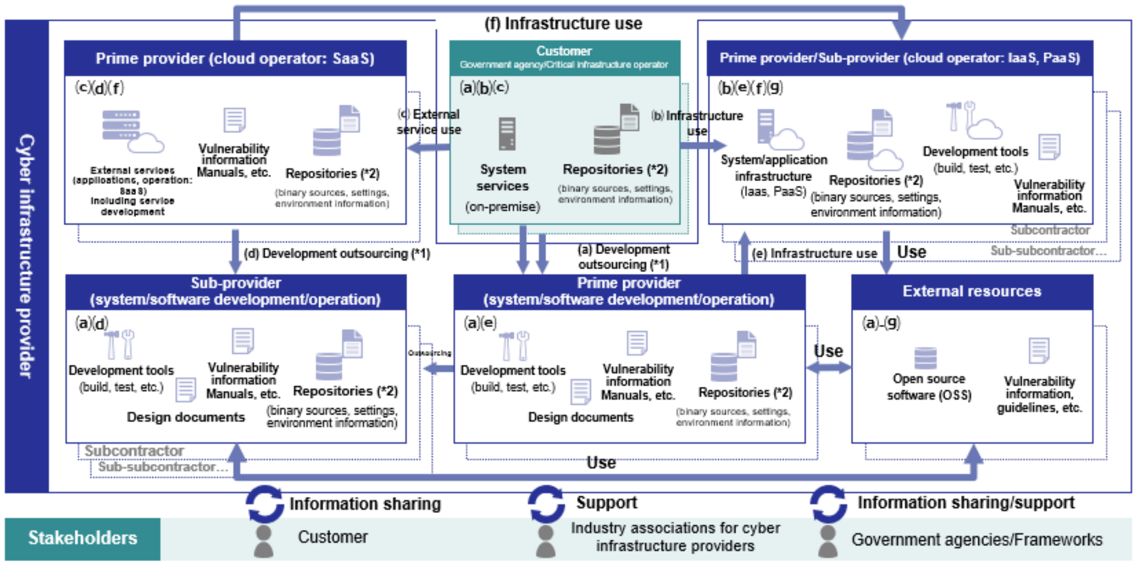
Figure 1 illustrates the relationships between the development, contract form, and usage form of software systems/assets handled by cyber infrastructure providers for which the Guidelines are intended. In this section, two roles described below are assumed for cyber infrastructure providers from the perspective of system development, contract, and usage:

[10] In some cases, developers and operators are also suppliers. In addition, in cases in which a sales company is also a cyber infrastructure provider, responsibilities equivalent to those of the supplier are required of them.
[11] Although it is usual that customers, who are the main entity of software utilization, operate software, specialized knowledge and skills are often required to operate systems and services or the software that composes them. In this context, it is assumed that cyber infrastructure providers support the operation of software (or part thereof) in accordance with contracts with customers.

- Prime provider
  A first-tier contractor that contracts directly with customers and develops, supplies, and operates systems and cloud services.

- Sub-provider
  A business that contracts with the prime provider and develops, supplies, and operates systems and cloud services[12].

The relationship between the prime provider and sub-provider is either a group company or an external contractor with no financial relationship. The supply chain of a subprovider may have a multi-tiered outsourcing structure, and each tier may form multiple hierarchical structures. In addition, external resources are public repositories of software, such as OSS, and these resources are operated by volunteer organizations that publish them, including information regarding vulnerabilities.


Cyber infrastructure provider
(f) Infrastructure use
Prime provider (cloud operator: SaaS)
(c)(d)(f)
External services (applications, operation: SaaS) including service development
Vulnerability information Manuals, etc.
Repositories (*2)
(binary sources, settings, environment information)
(c) External service use
Customer
Government agency/Critical infrastructure operator
(a)(b)(c)
System services (on-premise)
Repositories (*2)
(binary sources, settings, environment information)
(b) Infrastructure use
Prime provider/Sub-provider (cloud operator: IaaS, PaaS)
(b)(e)(f)(g)
System/application infrastructure (Iaas, PaaS)
Repositories (*2)
(binary sources, settings, environment information)
Development tools (build, test, etc.)
Vulnerability information Manuals, etc.
Subcontractor
Sub-subcontractor...
(d) Development outsourcing (*1)
(a) Development outsourcing (*1)
(e) Infrastructure use
Use
Sub-provider (system/software development/operation)
(a)(d)
Development tools (build, test, etc.)
Design documents
Vulnerability information Manuals, etc.
Repositories (*2)
(binary sources, settings, environment information)
Subcontractor
Sub-subcontractor...
Outsourcing
Prime provider (system/software development/operation)
(a)(e)
Development tools (build, test, etc.)
Design documents
Vulnerability information Manuals, etc.
Repositories (*2)
(binary sources, settings, environment information)
Use
External resources
(a)-(g)
Open source software (OSS)
Vulnerability information, guidelines, etc.
Use
Information sharing
Support
Information sharing/support
Stakeholders
Customer
Industry associations for cyber infrastructure providers
Government agencies/Frameworks

*1 There are cases where subcontractors are not used in (a) and (d). There are also cases where a customer conducts development for itself as a cyber infrastructure provider.

*2 The ownership of software varies depending on the service and system development, form of contract, etc., but there is no mention about the ownership.

[Examples of forms of system development, operation, and use]
(a): On-premise system (development outsourcing to prime provider)
(b): Customer uses IaaS, PaaS
(c): Use of system using SaaS
(d): Cloud provider (SaaS) outsources development
(e): Prime provider (SIer, etc.) uses IaaS, PaaS
(f): Cloud provider (SaaS) uses IaaS, etc.
(g): Cloud providers (IaaS, PaaS) have interoperation

Figure 1 Conceptual diagram of the parties involved in a software system composed of software

(4) Assumed risks

Software-related cybersecurity risks for which the Guidelines are intended refer to degrees of concern regarding security management, such as the leakage, loss, or damage of electromagnetic information owing to malicious attacks on software and defects in development or configuration errors, or degrees of concern regarding maintenance and administration, such as a decrease in safety and reliability or stoppage owing to attacks on systems or networks that handle electromagnetic information, and defects in development

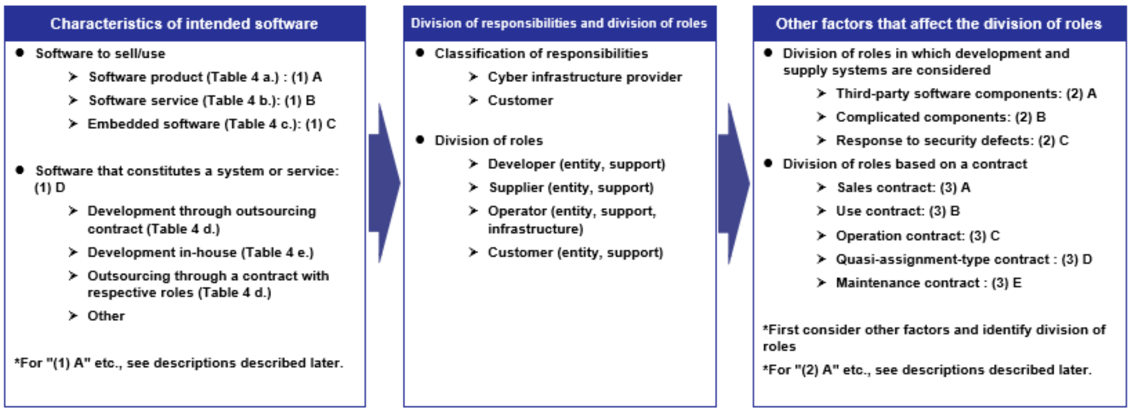
[12] The term "contract" is not generally used between SaaS providers, but for notation purposes, the term "contract" is used for both software development and services in this section.

including design or setting errors. There are various factors that cause cybersecurity risks, and they can become apparent at different stages of the software life cycle—from the analysis/planning phase of software products/systems and services to requirements definition, design, development, testing, release, operation, and disposal. These factors include insufficient risk analysis in the analysis/planning phase; insufficient agreement on security requirements in the requirements definition phase; insertion of unauthorized code or components in the development phase; insufficient reviews; tampering in the software distribution phase; service outages during operation; distribution of fake software and services by third parties impersonating legitimate businesses; insufficient preparation of people, things, and cost concerns during all phases; and insufficient management of supply chains. The Guidelines assume threats related to software spoofing, tampering, repudiation, information leakage, denial of service, and privilege escalation in all software phases.

1.4. Approach to division of roles

During software life cycle management, it is important to determine the responsibilities and division of roles of the respective parties concerned based on the characteristics of the software, software development/supply system, and contractual form of software use, operation, and development, as well as to promote the response to cybersecurity risks customers (the main entity of software utilization) face.. This section presents an example of the division of major roles in each intended scope of a software and describes a typical approach to the division of roles.

The Guidelines classify "**cyber infrastructure provider**" and "**customer**" based on responsibilities of these entities in the supply and use of software, respectively. Cyber infrastructure providers are classified into developer, supplier, and operator. If a business entity deserves to identify classification of respective responsibilities and division of roles, it is necessary to understand the position and scope of the roles, keeping in mind the characteristics of the intended software (intended scope of software, policy for the division of respective roles, etc.), as presented in Figure 2. Further, the classification of responsibilities and division of roles are identified based on the structural position of the intended software and the division of roles with other related development/supply systems or the roles stipulated under the contract.



Characteristics of intended software

- Software to sell/use
  - Software product (Table 4 a.) : (1) A
  - Software service (Table 4 b.): (1) B
  - Embedded software (Table 4 c.): (1) C
- Software that constitutes a system or service: (1) D
  - Development through outsourcing contract (Table 4 d.)
  - Development in-house (Table 4 e.)
  - Outsourcing through a contract with respective roles (Table 4 d.)
  - Other

*For "(1) A" etc., see descriptions described later.

Division of responsibilities and division of roles

- Classification of responsibilities
  - Cyber infrastructure provider
  - Customer
- Division of roles
  - Developer (entity, support)
  - Supplier (entity, support)
  - Operator (entity, support, infrastructure)
  - Customer (entity, support)

Other factors that affect the division of roles

- Division of roles in which development and supply systems are considered
  - Third-party software components: (2) A
  - Complicated components: (2) B
  - Response to security defects: (2) C
- Division of roles based on a contract
  - Sales contract: (3) A
  - Use contract: (3) B
  - Operation contract: (3) C
  - Quasi-assignment-type contract : (3) D
  - Maintenance contract : (3) E

*First consider other factors and identify division of roles

*For "(2) A" etc., see descriptions described later.

Figure 2 Factors affecting the classification of responsibilities and division of roles

Table 4 lists examples of assumptions for the respective cyber infrastructure providers and customer based on intended software characteristics. It also indicates corresponding roles in respective categories of responsibilities with a check mark. "Entity" and "Support" are added where it is generally expected that roles will be divided by the position of the main entity and its supportive position. In addition, "Infrastructure" is added to the role of providing the foundation on which a system operates. Note that, in the approach for the division of respective roles below, in cases in which assumed related operators play the role of both "customer" and "operator", they are mentioned separately by role. Even when a customer is a related business operator, if it has a department or person in charge whose role is "operator," it is considered that it shall have the responsibilities equivalent to "operator" as a cyber infrastructure provider. In addition, if the customer conducts development and supply in-house, it is considered that the customer itself will take on the responsibilities of "developer" and "supplier," which are the respective roles it shall take as a "cyber infrastructure provider" in the category of responsibilities, as the "(Entity)."

Table 4 Assumption of related operators and examples of classification of responsibilities/roles

Assumption of related operators		Division of roles / Classification of responsibilities	Developer	Supplier	Operator	Customer
a. Software product						
Software development vendor		**Cyber infrastructure provider**	✓			
Sales company		**Cyber infrastructure provider**		✓		
Purchaser (person in charge of operation)		**Cyber infrastructure provider**			✓	
Purchaser (user)		**Customer**				✓
b. Software service (where inter-service linkage is included)						
Service provider (prime provider)		**Cyber infrastructure provider**	✓ (Entity)	✓	✓	
Service provider (sub-provider)		**Cyber infrastructure provider**	✓ (Support)			
Service development support (sub-provider)		**Cyber infrastructure provider**	✓ (Support)			
Infrastructure operator (sub-provider)		**Cyber infrastructure provider**	✓			
Service user (person in charge of application operation)		**Cyber infrastructure provider**			✓	
Service user (application user)		**Customer**				✓
c. Embedded software						
Device development vendor		**Cyber infrastructure provider**	✓			
Embedded software development department		**Cyber infrastructure provider**	✓			
Sales company		**Cyber infrastructure provider**		✓		
Purchaser (person in charge of operation)		**Cyber infrastructure provider**			✓	
Purchaser (user)		**Customer**				✓
d. System (system owner makes plans and procures development/operation/infrastructure services)						
Development operation contractor		**Cyber infrastructure provider**	✓ (Entity)	✓ (Entity)	✓ (Support)	
Development support		**Cyber infrastructure provider**	✓ (Support)	✓ (Support)		
Software component development		**Cyber infrastructure provider**	✓	✓		
Infrastructure operator (IaaS/PaaS)		**Cyber infrastructure provider**	✓	✓ (Infrastructure)	✓ (Infrastructure)	
Procurer (system operator)		**Cyber infrastructure provider**			✓ (Entity)	
Procurer (system owner)		**Customer**				✓
e. System (in-house development, affiliated operator supports development/supply/operation)						
Parent operator (development department)		**Cyber infrastructure provider**	✓ (Entity)	✓ (Entity)		
Affiliated operator		**Cyber infrastructure provider**	✓ (Support)	✓ (Support)	✓ (Support)	
Parent operator (operation department)		**Cyber infrastructure provider**			✓ (Entity)	
Parent operator (user department)		**Customer**				✓
f. System (example of a case in which the user departments, operation department, and development department of a business that is the customer take on respective roles as the (entity) and outsource part of the tasks of the respective roles (support) to another business as a procurer severally under a quasi-delegation contract)						
Procurer (development department)		**Cyber infrastructure provider**	✓ (Entity)	✓ (Entity)		
Procurer (operation department)		**Cyber infrastructure provider**			✓ (Entity)	
Procurer (user department)		**Customer**				✓ (Entity)
Consultation (systemization concept)	Case	**Customer**				✓ (Support)
Research company (PMO support)	Case	**Customer**				✓ (Support)
Development vendor (development)	Case	**Cyber infrastructure provider**	✓ (Support)	✓ (Support)		
Operation vendor (operation/maintenance)	Case	**Cyber infrastructure provider**			✓ (Support)	

(1) Approach to division of roles by software characteristics

The approach to the division of roles by the software characteristics is described below.

a. Software product

In case of a software product, the developer and customer are different businesses, and the supplier acts as the software product sales agent or the software is sold directly by the developer (the developer also serves as the supplier). The operator of the software product is typically the customer who uses the software product or the customer's operations department.

b. Software service

In case of a software service, the developer and customer are different businesses, and the service provider serves as the developer, supplier, and operator of the service. When a service user (customer) configures and runs applications on its own terms using the software service as a platform (for example, when using a cloud service), as an operator, it is common to determine the scope of responsibilities of the respective stakeholders based on the concept of shared responsibility. In this case, the responsibility of operation is shared; for example, the service provider is responsible for operating the system built on the infrastructure and the service user is responsible for the operation of applications.

c. Embedded software

In case of an embedded software, assuming that it is sold and used with the device in which the software is embedded, the developer is generally considered to be the device developer who possesses software development departments. The operator of a device with the embedded software is typically the customer or the operation department of the customer.

d. Software service that constitutes a system service

In case of a software that constitutes a service system (for example, a business), the main entity that uses or provides the service (generally known as the system owner) plays the role of the customer in the Guidelines. For the development, supply, and operation of such service systems, it is assumed that, depending on the system's scale and the specialized knowledge and skills required, these responsibilities may be undertaken by a group of businesses other than the system owner. In addition, it is assumed that a multi-layered outsourcing structure will be established. This structure may include roles such as a prime provider, sub-provider, cloud operator (providing the infrastructure environment), and multiple hierarchical layers within each role (see Figure 1). In such a case, it is necessary to determine the division of the respective roles based on the components of the system service, development/supply process, and operation process system. The division must consider the hierarchical structure involving a developer, a supplier, an operator, and their mutual cooperation. In terms of the operation of an IT system, it is common for the operation department of the system owner (customer) to be responsible for the overall role of the operator or for coordinating operations with external or outsourced businesses to share the role of the operator across the entire operation system.

(2) Approach to division of roles within software development/supply system

Examples of the approach to the division of roles within software development/supply systems are described below.

a. Division of roles for development/supply when third-party software components are included

When software components include third-party software components, the third party is positioned as a business that participates in the roles of both the component developer and the component supplier to the developer using those components.

b. Division of roles for development/supply when software has a complex mix of components

Cases in which there is a complex mix of software components with multiple third-party software components, the component structure of such software may become hierarchically complex. In systems and services, multiple software components with complex, hierarchical structures will be further integrated into the overall configuration. Even for a software system that runs multiple components in combination and there is a specific developer responsible for each component, there are developers who take combined responsibility for the entire software system. All businesses involved in the development of such software are expected to recognize their responsibilities as developers (and suppliers of the components that they are in charge of) according to the Guidelines and fulfill their specified roles. The principle is that at least all software components should be in a state in which the responsibility of the developer is held by one of the businesses. Under this development/supply system, one should establish a system for software development, supply, and defect correction and allocate roles appropriately.

c. Division of roles in development/supply related to response to security defects

At the point of contact for customers who are the primary users of software, the supplier is responsible for securely releasing the software that has been tested by the developer. The developer, in turn, is responsible for the processes related to defect correction; this includes providing contact points for receipt of notifications when a security defect (which may include vulnerabilities) is discovered during operations, and issuing security advisories.

(3) Approach to division of roles by contract type for software use, operation, and development

Examples of the approach to the division of roles by contract type for software use, operation, and development are described below.

a. In the case of a product purchased through a sales contract

In principle, the customer is responsible for the use and operation of a software product purchased by the customer. License and maintenance agreements for software use are established between the customer and software supplier. In addition, a sales contract is concluded between the developer and supplier (mainly a seller), and the division of roles for the software sales rights and maintenance is stipulated.

b. In the case of a service obtained through a usage contract

When a customer uses a software service (such as a cloud service) provided by a service provider, customer uses the service, and outsources operations of the service to the service provider. For the operational part, in particular, the extent to which the customer conducts the operation proactively and responsibly, the extent to which part of the operational responsibilities of the service is to be entrusted to the service provider, and the demarcation point between these extents are identified based on the concept of a responsibility-sharing model. Thereafter, a usage contract is established based on the terms of use and service level agreement (SLA).

c. In the case of operation outsourcing through an operation contract

When a customer outsources all or part of the operation of a system, including the software, to a cyber infrastructure provider, they enter into an operation contract (including a maintenance agreement, if necessary) with the cyber infrastructure provider who will be the operator and share the roles for system operation, including the software, based on the contract.

d. In the case of service outsourcing through a quasi-assignment-type contract

When a customer outsources software development, they enter into a work-contract-type or quasi-assignment-type software development outsourcing contract with a cyber infrastructure provider (development/supply) who serves as the contact point. In many cases, a work-contract-type contract is concluded when development specifications, including security requirements, are created and the completion responsibility for system development, including software development (design, programming, testing, installation, implementation, data migration, training, and preparation for release), is imposed on the customer side. In contrast, a quasi-assignment-type contract may be employed for service provision at the specification examination stage, such as a systemization concept. In addition, services for development and operation may be received through quasi-assignment-type contracts. In a work-contract-type development outsourcing contract, the outsourcing business assumes the roles of developer and supplier. In the case of a quasi-assignment-type outsourcing contract, it is advisable to determine the scope of the roles in which service provision is made, recognize the division of roles and responsibilities corresponding to the roles, and clearly identify the implementation content corresponding to the responsibilities in the contract.

e. Division of roles in software development/operation through a maintenance contract

Following the completion of system development, including software development, the customer will accept the developed system, including the software, and conclude a maintenance contract with the developer, which will include terms such as responding to software defects when starting to operate the system; it is necessary to determine liability for non-compliance with the development contract separately.

A maintenance contract generally includes responses to inquiries, investigation of defects, and responses to defect corrections based on regulations (or provision of updated software). When selecting a software maintenance contract, the form of contract appropriate for the actual work and service content related to the maintenance must be selected. In the case of a contract that primarily involves information and version provision, upgrades are primarily made from the development/supply side, and the operation department on the customer side takes charge of the application of the provided information and updates. However, in the case of maintenance, which includes responding to inquiries regarding the software (such as how to use it, unclear points, confirmations, and questions about technical issues) and defect corrections within a specified scope, a quasi-assignment contract is usually established. On the contrary, when the responsibility for the completion of repair in response to software defects is to be attached, a work-contract-type contract is preferred.

In the Guidelines, it is assumed that a maintenance contract for a software product includes the developer's responsibility to respond to vulnerabilities, including defect correction. For software whose development is outsourced, operations such as the conclusion of a maintenance contract equivalent to a work contract (including the developer's responsibility to respond to vulnerabilities), the conclusion of a work-contract-type maintenance contract or quasi-assignment-type maintenance contract, and the conclusion of memorandums of understanding regarding changes to specifications and costs upon agreement, are assumed.

1.5. Examples of typical use cases

In software life cycle management, the responsibilities and division of roles of the respective parties concerned are determined based on the characteristics of the software, software development/supply system, and contractual form of the software use and operation; in addition, it ensures prompt responses to cybersecurity risks facing the main entity of software utilization—the customer. In this section, the division of roles among multiple cyber infrastructure providers are presented and described for the following four use cases:

- Use case example of roles in a software product and embedded software
- Use case example of roles in a software service
- Use case example of roles in a system developed through outsourcing contract
- Use case examples of roles in a system developed in-house

[1] Use case example of roles in a software product and embedded software

As a use case example of the development and supply of a software product, the case of a customer (purchaser) who procures a software product is described (see Figure 3). The customer purchases a software product from a sales company; in response to the order (or as procurement), the sales company places an order for the software product with a software development vendor. While the software development vendor (prime provider) takes charge of the development, commercialization, and shipment of the software product, an external software development company (sub-provider) is responsible for the development of software components.

As a use case example of the development and supply of IoT devices (with embedded software), a case of a customer (purchaser) who is a procures an IoT device with an embedded software is described. The IoT device is procured from a sales company, which places an order for the IoT device with a device development vendor in response to the order (or as procurement). While the device development vendor is responsible for the development of the IoT devices, implementation of the embedded software, commercialization, and shipment, the embedded software development department of the device development vendor develops the embedded software.

For example, if the software development vendor that provides the software product uses SaaS provided by a cloud operator as the system infrastructure, when providing software update services via the cloud, the cloud operator assumes the roles of provider and developer/operator. In addition, if any business uses external resources, the developer, provider, or operator manages them appropriately, depending on the form of use.

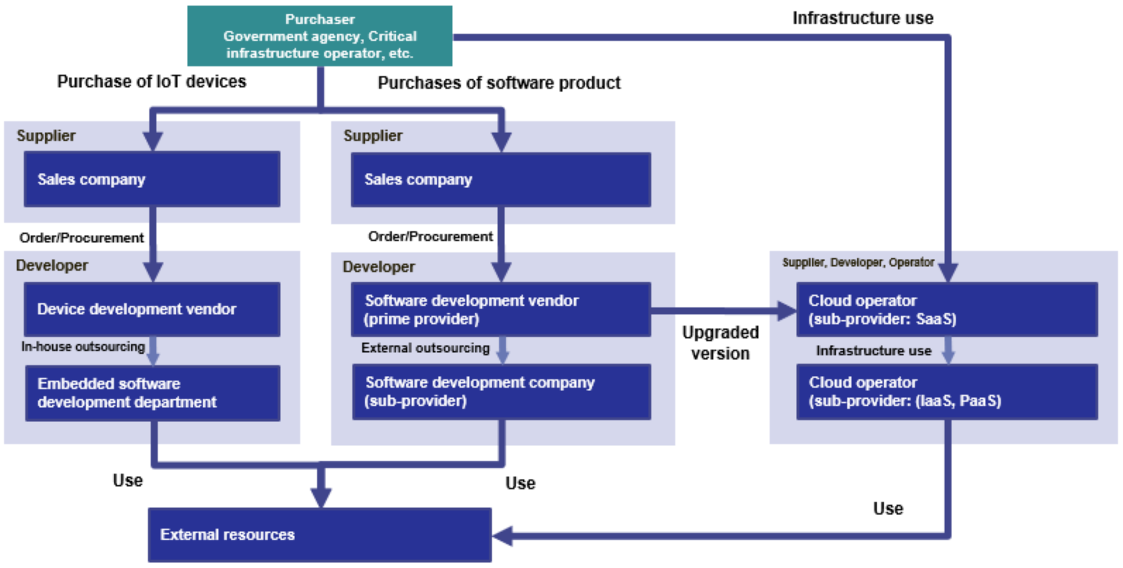

Purchaser
Government agency, Critical infrastructure operator, etc.
Infrastructure use
Purchase of IoT devices
Purchases of software product
Supplier
Sales company
Supplier
Sales company
Order/Procurement
Order/Procurement
Developer
Device development vendor
In-house outsourcing
Embedded software development department
Developer
Software development vendor (prime provider)
External outsourcing
Software development company (sub-provider)
Upgraded version
Supplier, Developer, Operator
Cloud operator (sub-provider: SaaS)
Infrastructure use
Cloud operator (sub-provider: (IaaS, PaaS)
Use
Use
Use
External resources

Figure 3 Conceptual diagram of use case example of the roles in a software product with embedded software

[2] Use case example of roles in a software service

Here, an example of a use case in which a cloud service that employs SaaS is described (see Figure 4). In this use case, a service provider (the prime supplier) supplies a SaaS service and takes on the developer and operator roles; a service development business or a sub-provider takes responsibility of developing the software that constitutes the SaaS service, and the same or a different cloud operator takes responsibility of the supply, development, and operation of an IaaS service that runs the SaaS service (see Figure 4(a) "Infrastructure operator (sub-provider: IaaS, PaaS)"). In addition, when the cloud operator or sub-provider uses external resources, the developer, provider, or operator manages them appropriately, depending on the form of use.

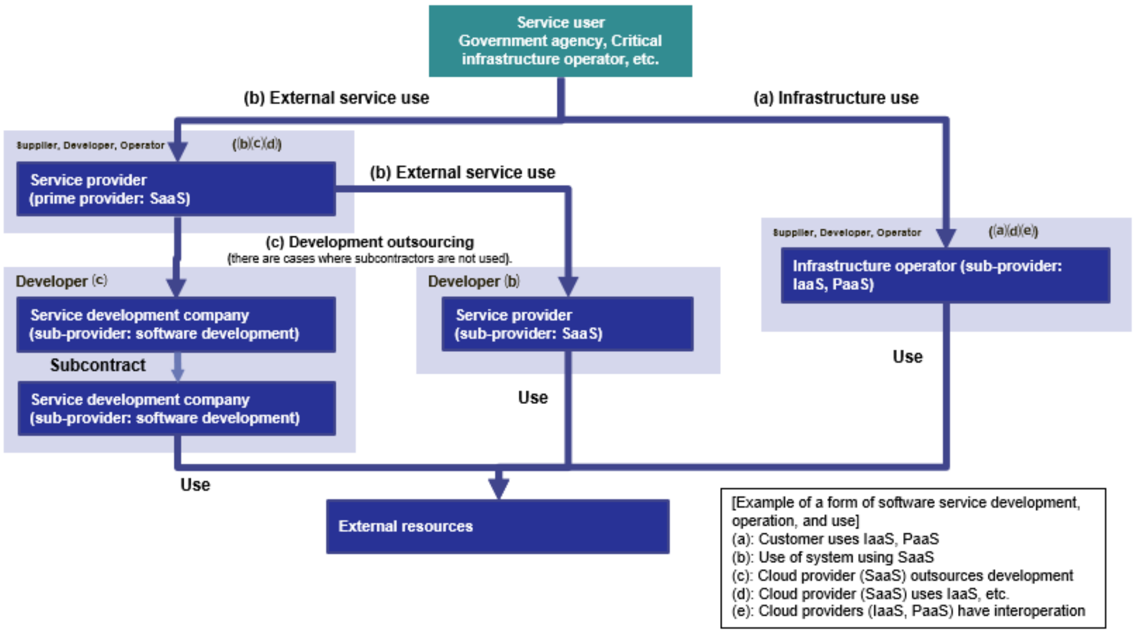

Service user
Government agency, Critical infrastructure operator, etc.
(b) External service use
(a) Infrastructure use
Supplier, Developer, Operator
(b)(c)(d)
Service provider (prime provider: SaaS)
(b) External service use
(c) Development outsourcing
(there are cases where subcontractors are not used).
Supplier, Developer, Operator
(a)(d)(e)
Infrastructure operator (sub-provider: IaaS, PaaS)
Developer (c)
Service development company (sub-provider: software development)
Subcontract
Service development company (sub-provider: software development)
Developer (b)
Service provider (sub-provider: SaaS)
Use
Use
Use
External resources
[Example of a form of software service development, operation, and use]
(a): Customer uses IaaS, PaaS
(b): Use of system using SaaS
(c): Cloud provider (SaaS) outsources development
(d): Cloud provider (SaaS) uses IaaS, etc.
(e): Cloud providers (IaaS, PaaS) have interoperation

Figure 4 Conceptual diagram of use case example of roles in a software service

[3] Use case example of roles in a system developed through outsourcing contract

An example of a use case is described in which an outsourcing contract is created for the development of an IT system, which is commonly observed in the procurement of the design, development, operation, and maintenance of a business system that employs the government cloud (see Figure 5). This example describes how an IT system is developed and deployed, and the support and substitution for the operation and maintenance work of the IT system are procured. The prime provider, which is a SIer, acts as the supplier while taking on the roles of developer and operator (see in Figure 5(a), development outsourcing contractor: "prime provider: system and software development and operation") and the sub-provider undertakes part of the development or develops and manufactures software products and IoT products that are components of the IT system (such as the "sub-provider: system and software development", which refers to the contractor and its subcontractor).

If a contract is created for PaaS of the cloud operator via the prime provider—wherein the cloud operator acts as a sub-provider—and the PaaS is used as the system infrastructure ((b) Infrastructure usage: "Infrastructure provider (sub-provider: IaaS, PaaS)" in the figure), the infrastructure operator takes on the roles of developer and operator. In addition, when using external resources, the developer, supplier, or operator must have proper administration, depending on the form of use.

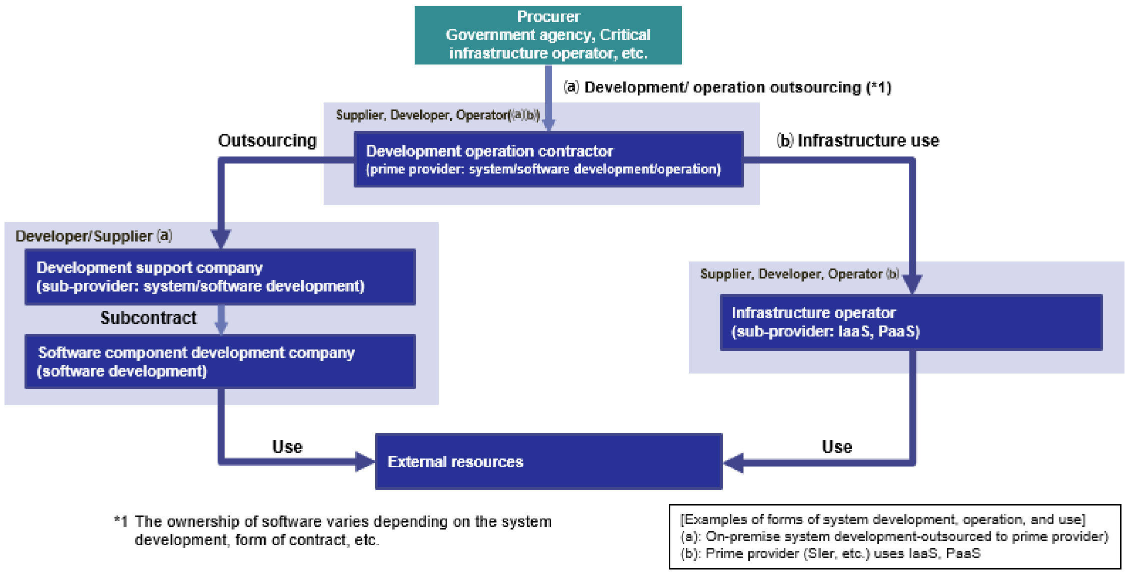

Procurer
Government agency, Critical infrastructure operator, etc.
(a) Development/ operation outsourcing (*1)
Supplier, Developer, Operator((a)(b))
Outsourcing
Development operation contractor
(prime provider: system/software development/operation)
(b) Infrastructure use
Developer/Supplier (a)
Development support company
(sub-provider: system/software development)
Subcontract
Software component development company
(software development)
Supplier, Developer, Operator (b)
Infrastructure operator
(sub-provider: IaaS, PaaS)
Use
External resources
Use
*1 The ownership of software varies depending on the system development, form of contract, etc.
[Examples of forms of system development, operation, and use]
(a): On-premise system development-outsourced to prime provider)
(b): Prime provider (SIer, etc.) uses IaaS, PaaS

Figure 5 Conceptual diagram of a use case example of roles in a system developed through outsourcing contract

[4] Use case example of roles in a system developed in-house

An example of a use case is described wherein a business develops an IT system for its in-house use (see Figure 6). In some cases, businesses have a development department and an operation department to support the operation of their IT system to be used in the user department (corresponding to the customer in the Guidelines). It is expected that the responsibilities and division of roles in such cases will be assigned to the user department (customer), development department (developer), and operation department (operator).

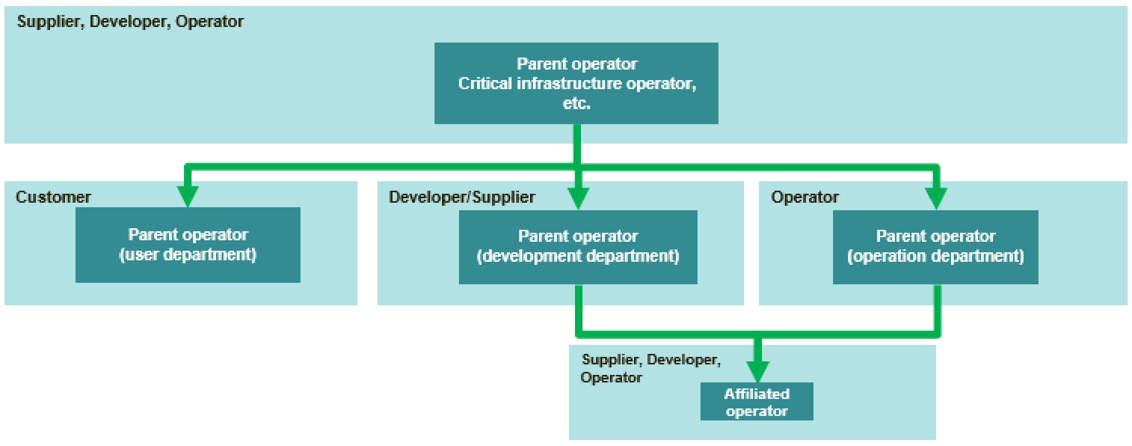


Figure 6 Conceptual diagram of a use case example of roles in a system developed in-house development

2. Responsibilities and division of roles of cyber infrastructure providers and customers

2.1. Approach to responsibilities and division of roles

The extend to which a single cyber infrastructure provider can solely reduce security risks in a software supply chain is limited. Therefore, it is necessary for cyber infrastructure providers that make up the supply chain to coordinate with their customers individually or in cooperation while fulfilling their respective responsibilities. For example, in the requirements definition phase, along with the cyber infrastructure provider who performs appropriate risk analysis, the customer also has the obligation for risk management of the entire system owned by the customer. If risks are not promptly identified, it will be difficult to evaluate security requirements, causing software vulnerabilities to remain hidden.

That is, the customer, under the leadership of the management, must clarify the division of roles with the cyber infrastructure provider regarding risk management for its own in-house systems, present security requirements to the cyber infrastructure provider so that it can identify items on which it must make decisions and adjustments as the software product/service user, and purchase appropriate products and maintain a system for evaluating the quality of the results of work that it commissioned in-house.

In addition, the cyber infrastructure provider has an obligation to take security measures for its own products and services, and it can be stated that the management is required to take the lead in promoting measures so as not to place security responsibilities solely on the customer.

These concepts are summarized as responsibilities in the following sections.

2.2. Responsibilities

To improve cybersecurity-related resilience, complementary effects can be obtained when cyber infrastructure providers and customers fulfill their respective responsibilities.

<Responsibilities of cyber infrastructure providers>

Cyber infrastructure providers must be aware of the following five responsibilities to improve cybersecurity resilience. All of these responsibilities must be recognized by the management of each cyber infrastructure provider, and efforts to fulfill these responsibilities must be implemented under the leadership of the management.

(1) Design, development, supply, and operation of software with security quality ensured

- **Providing secure software and evaluating measures**
  In accordance with the principles of "secure by design" and "secure by default," take measures to reduce threats to software development and operation in accordance with a risk-based approach, and determine their effectiveness. In addition, enforce minimum security standards for the software.

- **Consideration of cybersecurity throughout the entire software life cycle**
  Starting with an agreement on security requirements, consider cybersecurity throughout the entire software life cycle agreed upon with the customer, including secure build, testing, and operation.

(2) Software supply chain management

- **Sharing of implementation status of security control measures**
  To allow users to make decisions regarding software procurement and implementation—including the selection of risk-based solutions—suppliers should disclose the status of their software development efforts.
  Ensure transparency with customers regarding all necessary aspects of cybersecurity.

- **Sharing of software configuration information**
  For measures against vulnerabilities by users, use information from software configuration management, including the software bill of materials (SBOM), and configuration information, including OSS.

- **Promotion of risk management including supply chains**
  Include suppliers (such as system integrators, external system service providers, and partners), developers, and all other businesses related to IT/OT/ICT systems in the scope of software supply chain risk management activities.

(3) Prompt response to remaining vulnerabilities

- **Communication and response to vulnerabilities and threat information**
  Arrange vulnerability disclosure policies appropriately and establish a vulnerability response system. Developers, suppliers and operators are responsible for identifying and disclosing vulnerabilities in cloud service software, providing the information necessary for secure service configuration and operation, upgrading services, developing and distributing patches, and documenting upgrade/patch application processes so that customers understand how to participate in the processes. In addition, maintain a mechanism for sending notifications to customers.

(4) Arrangement of governance for software

- **Integration of software supply chain risk management into enterprise risk management**
  Software supply chain risk management covers activities throughout the software life cycle and is as part of the enterprise risk management process.
  Arrange the resources necessary to reduce risk to an acceptable level (people, materials, and money) in your organization. Position cybersecurity as a key management issue, and the top management must be made responsible for implementing risk management.
  Comply with laws and regulations.

(5) Strengthening of information sharing and cooperation systems between cyber infrastructure provider and stakeholder

- **Sharing of threat and vulnerability information among stakeholders and response to it**
  Share threat and vulnerability information with government and industry partners in a prompt and timely manner. Suppliers must share software vulnerability information with the relevant agencies that have jurisdiction.

- **Collaboration among stakeholders engaged in cybersecurity**
  All stakeholders, including communities[13], must work together in a healthy manner to develop a framework for identifying potential risks and assessing supply chain risk dependencies related to cybersecurity.
  In terms of security measures, take initiative and share responsibilities throughout the entire supply chains, including platform providers and consumer tenant organizations.
  In cooperation with the government, the private sector must continually adapt to the necessary requirements and improve the security of the technologies, products, and services supporting businesses that provide critical infrastructure.
  Appropriate and timely participation of stakeholders enables sharing of knowledge and awareness, which leads to appropriate risk management.

[13] Examples of communities include ISACs (such as Software ISAC), CSIRT council, communities composed of volunteer groups formed by private companies, and regional security communities.

<Responsibilities of customers>

In activities related to software security that constitute a system in which a customer has ownership, the customer has the following responsibilities:

(6) Risk management and software procurement/operation by the leadership of the customer's management

- **Risk management by the leadership of the customer's management**
  Risk management with independent and proactive initiatives and cooperative measures by the customer based on a contract with a cyber infrastructure provider.
  Allocation and preparation of resources to respond to known vulnerabilities proactively and implement measures for mitigation.
  Proactive participation in and utilization of communities and cooperative systems aimed at security improvement among related parties, including customers and cyber infrastructure providers.

- **Software procurement/operation by the leadership of the customer's management**
  Presentation of security requirements to incorporate security functions into software design plans.
  Disclosure of requirements for security practices in software procurement/implementation.
  Decision-making based on risk assessment in software procurement/implementation.
  Budgeting for software operation, risk response, and contracts considering the life cycle

In activities based on the responsibilities of cyber infrastructure providers, specifically the activities associated with customers, it is important that customers are aware of their responsibilities and support the activities that fulfill these responsibilities based on reasonable agreements to contribute toward improving cybersecurity resilience.

3. Requirements for fulfilling responsibilities

3.1. Overview of the requirements

To fulfill their responsibilities toward improving cybersecurity resilience, cyber infrastructure providers and customers are required to implement the cybersecurity measures described below (six categories and 21 requirements) in a manner that is appropriate to the characteristics of the intended software and the organization. Therefore, under the leadership of the management responsible for risk management in the organization, it is necessary to proceed with the implementation policy of measures appropriate to the risks, allocation of budgets and human resources, confirmation of the implementation status, identification of problems, responses to problems, and cooperation with other related organizations.

For initiatives that are difficult to handle in-house or that are deemed appropriate for implementation by an expert business, it is necessary to consider outsourcing a part of such initiatives.

The requirements for improving cybersecurity resilience based on these approaches are described below. Note that the identification of respective requirements is in the form of “S(n1)-n2” (where n1 is the category number and n2 is the sequential number in the category), and the identification of respective itemized requirements of respective requirements are in the form of “S(n1)-n2.n3” (where n3 is the sequential number of the itemized requirement in the requirement).

<Requirements for cyber infrastructure providers>

(1) Secure design, development, supply, and operation

Develop, supply, and operate software that checks vulnerabilities and has security.

S(1)-1 Risk assessment during design and tracking of countermeasures

S(1)-2 Secure build

S(1)-3 Testing

S(1)-4 Monitoring of services

(2) Life cycle management and assurance of transparency

Provide an assurance of transparency in software management throughout the life cycle and manage risks including those in the supply chain.

S(2)-1 Arrangement of secure components

S(2)-2 Secure archiving of release files and data

S(2)-3 Establishment of security requirements among stakeholders

S(2)-4 Appropriate information provision to users

(3) Prompt response to remaining vulnerabilities

Identify vulnerabilities remaining in released software and respond to them promptly

S(3)-1 Continuous vulnerability investigation

S(3)-2 Responses to detected vulnerabilities

S(3)-3 Application of results of countermeasures to in-house process improvement

(4) Arrangement of human resources, processes, and technologies

Arrange human resources, processes, and technologies related to software at the organizational level

S(4)-1 Human resources: Commitment from management and arrangement of personnel

S(4)-2 Process: Establishment of development policy and compliance with laws and regulations

S(4)-3 Process: Establishment of operation policy and compliance with laws and regulations

S(4)-4 Process: Establishment of development and operational standards

S(4)-5 Technology: Arrangement of secure development tools

S(4)-6 Technology: Arrangement of secure development environments

(5) Strengthening of relationships between cyber infrastructure providers and stakeholders

Reinforce information sharing and cooperation between cyber infrastructure provider and stakeholders.

S(5)-1 Organizational system for information sharing

S(5)-2 Strengthening of cooperation systems

<Requirements for customers>

(6) Risk management by customers, and procurement and operation of secure software

Implement risk management, and secure software procurement and operation under the leadership of the customer's management

S(6)-1 Risk management under the leadership of the customer's management

S(6)-2 Software procurement and operation under the leadership of the customer's management

Figure 7 shows a conceptual diagram of the relationship between these six categories of requirements and a general system of security measures.

The outer rim indicates the responsibilities associated with the software lifecycle. The gradient illustrates the seamless continuity between phases of the software lifecycle and the integration and coordination of roles. This figure is used as a reference to explain the individual requirements in and after 3.2, highlighting the relevant categories explained in each section.

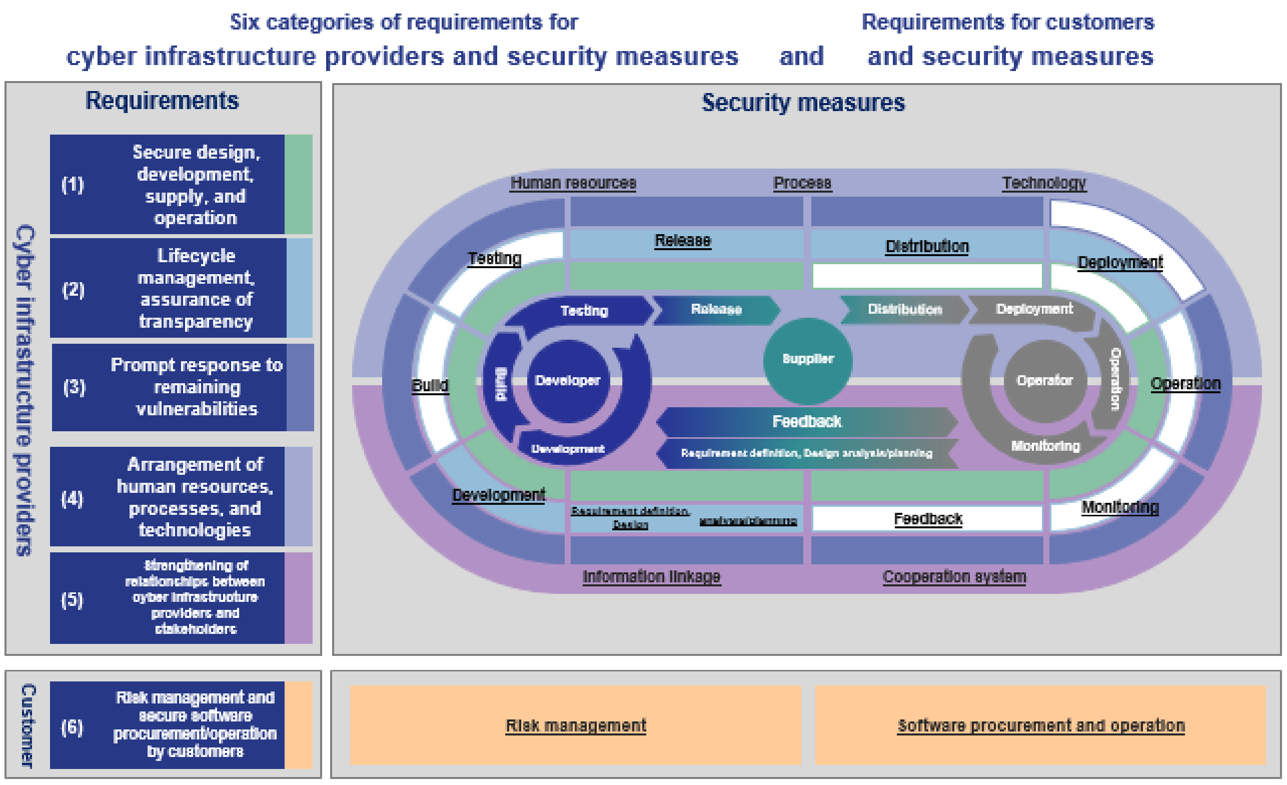

Six categories of requirements for
cyber infrastructure providers and security measures
and
Requirements for customers
and security measures
Requirements
Cyber infrastructure providers
(1) Secure design, development, supply, and operation
(2) Lifecycle management, assurance of transparency
(3) Prompt response to remaining vulnerabilities
(4) Arrangement of human resources, processes, and technologies
(5) Strengthening of relationships between cyber infrastructure providers and stakeholders
Customer
(6) Risk management and secure software procurement/operation by customers
Security measures
Human resources
Process
Technology
Testing
Release
Distribution
Deployment
Build
Operation
Development
Requirement definition, Design
analysis/planning
Feedback
Monitoring
Information linkage
Cooperation system
Developer
Supplier
Operator
Requirement definition, Design analysis/planning
Risk management
Software procurement and operation

Figure 7 Conceptual diagram of requirements

In the reference information, a checklist of requirements, practice examples, related reference information, and explanations of terms mentioned in the Guidelines are described.

3.2. Requirements

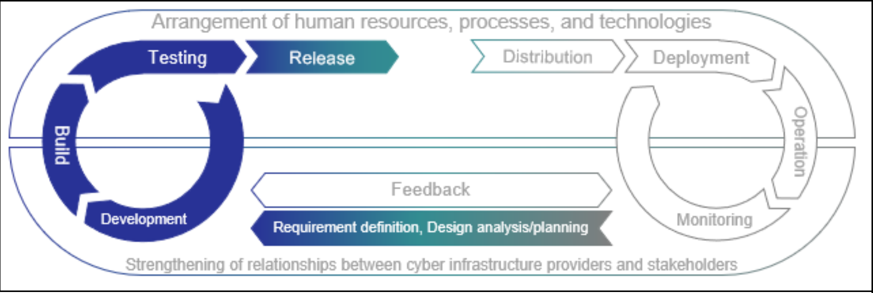
The requirements set out in the Guidelines are described with the following configuration:

- **Identification**
  To identify requirements, the category number is followed by "S," followed by the sequential number within the category, such as "(1)-1."

- **Requirement title, intended role, summary, and point relevant in the life cycle**
  The title of the requirement, the role for which the requirement is essential, and a summary of the title are provided.
  In the lower section, the stage to which the requirement applies in the software life cycle in the conceptual diagram of requirements above is indicated with "fill."

- **Itemized requirements**
  For the respective requirements, the contents of the itemized requirements that encourage the intended person to take specific measures are shown.

(1) Secure design, development, supply, and operation

S (1)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Development Feedback Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Risk assessment during design and tracking of countermeasures Analyze and assess the risks of software to be developed in accordance with the principles of "secure by design" and "secure by default"; track risk responses, security requirements, and design decisions; and maintain countermeasures.		

Itemized requirements

☐ **S(1)-1.1 Risk-based security requirements definition**
Perform risk-based analysis and assessment of the software to be developed or the system/service using the software, and define security requirements that serve as mitigation measures.

☐ **S(1)-1.2 Design review**
Through a review of the software design, confirm that it meets all security requirements and adequately addresses identified risks, and apply the review results.

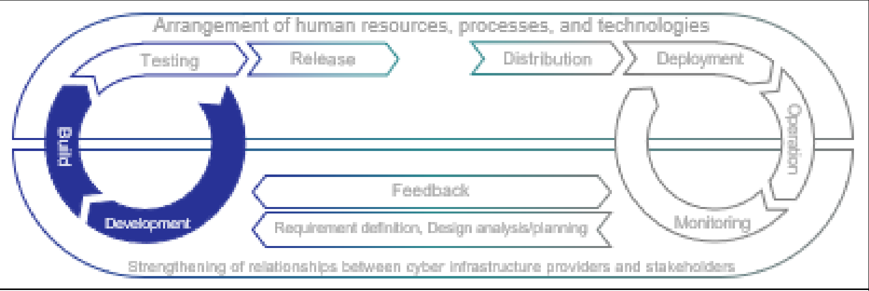
☐ **S(1)-1.3 Risk response records**
Maintain records of design decisions, responses to risks, and approved exceptional measures for audit and maintenance purposes throughout the software life cycle.

☐ **S(1)-1.4 Periodic risk-based review**
Review all approved exceptions to security requirements and software design, as well as the results of the risk-based analysis and assessment created during the software design, and periodically check whether risks are being addressed appropriately.

S(1)-1 requires software developers to design software that meets security requirements and mitigates security risks.

When security requirements have already been identified, reviewing the software design and verifying that it adequately addresses security requirements and risks helps to ensure that the software satisfies the security requirements and can fully respond to the identified risk information. Responding to security requirements and risks in the software design stage (secure by design) and embedding software security by default (secure by default) are key factors in improving software security and improving development efficiency.

To derive software security requirements, a risk-based analysis is required to identify and evaluate them. Security risks that may be faced during the operation of the software, and how these risks should be mitigated with the software design and architecture should be determined. In addition, determining whether security requirements should be relaxed or waived through a risk-based analysis helps to prove its validity.

S (1)-2	**Developer**	
	Supplier	
	Operator	
	Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Development
Requirement definition, Design analysis/planning
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Secure build
Define secure coding and system construction processes that are appropriate for development languages and development environments, and generate and build code accordingly. Review and analyze the code, including configurations, and feed the results back to the process.

Itemized requirements

☐ **S(1)-2.1 Definition of secure development process**
Define processes related to secure coding, secure build, and secure by default by considering secure coding perspectives, the build timing and method, the use of automation tools, and training.

☐ **S(1)-2.2 Secure build**
Generate and build code using a compiler, an interpreter, and build tools that provide functions to improve the security of executable formats.

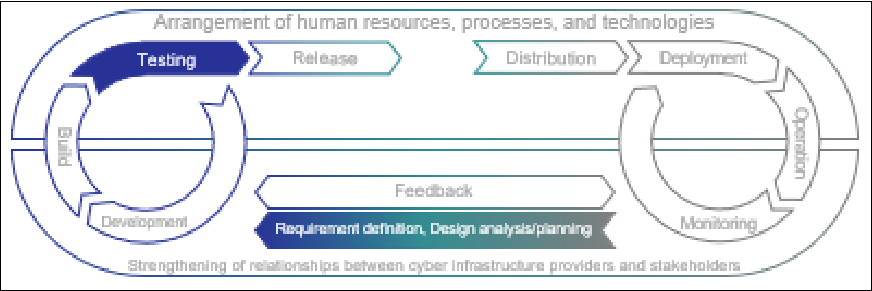
☐ **S(1)-2.3 Verification and feedback**
Identify root causes of problems discovered through verification by review and analysis, and then feed the results back to the processes.

☐ **S(1)-2.4 Codebases**
For objects subject to review and analysis, not only source codes but also codes in various formats (such as configuration files) that the organization determines to be readable should be targets.

S(1)-2 requires software developers to generate and build software codebases securely.

Adhering to secure coding practices and generating source codes and codebases with secure configurations reduce software security vulnerabilities. In addition, for vulnerabilities included in the codebase generation, applying processes to ensure that they are below the vulnerability tolerance levels defined by the organization or minimizing those that exceed the levels leads to a reduction in costs. To improve the security of executable formats, establishing compile, link, and build processes to eliminate vulnerabilities before testing reduces security vulnerabilities in software and also leads to a reduction in costs. Reviewing and analyzing code enables compliance with the security requirements to be verified. In addition, when vulnerabilities are identified during the process, they can be fixed before software release to prevent exploitation.

Applying automated measures to these codebase generation and build processes can reduce the efforts and resources required to detect vulnerabilities.

S (1)-3	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Development Feedback Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Testing
Design and implement vulnerability testing and penetration testing as well as functional testing to find vulnerabilities not identified in the review and analysis up to the build phase, and subsequently take countermeasures against identified vulnerabilities.

Itemized requirements

☐ **S(1)-3.1 Test planning**
Based on threat models and risk analysis, determine a test scope and test method, and develop a test plan.

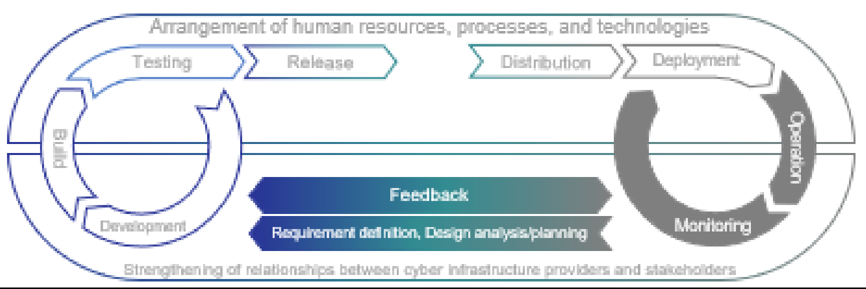
☐ **S(1)-3.2 Test method**
Include functional testing, vulnerability testing, fuzzing, penetration testing, etc. in the test method.

☐ **S(1)-3.3 Test implementation**
Design and implement tests according to the test plan, and document the test results.

☐ **S(1)-3.4 Responses to problems**
Incorporate all problems identified through testing and recommended countermeasures into the development team's workflows to solve them.

S(1)-3 requires software developers to find and respond to vulnerabilities through testing.

Testing an executable code can verify compliance with security requirements. In addition, when vulnerabilities are identified during the process, they can be fixed before the software release to prevent exploitation. By applying automated methods to the testing process and arranging appropriate evidence and environments according to the form of implementation, it is possible to reduce the efforts and resources required to identify vulnerabilities and improve traceability and reproducibility. Note that with respect to the testing method, the policy varies depending on whether the intended software is a product or service developed in-house, a system or service developed on a contract basis, or a development method (waterfall or agile development). Based on security requirements defined on a risk basis and the defined secure development process, a policy for the testing method should be determined and a test plan created.

S (1)-4	**Developer**	
	Supplier	
	Operator	
	Customer	
Monitoring of services Arrange a process and system that monitors software protects and maintains information assets and is consistent with the environment in which it is implemented (network, platform, service, etc.), and implement these.		

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Development
Monitoring
Feedback
Requirement definition, Design analysis/planning
Strengthening of relationships between cyber infrastructure providers and stakeholders

Itemized requirements

☐ **S(1)-4.1 Asset management**
Operators arrange asset management procedures and asset lists related to assets handled by systems and services as well as assets that constitute the systems and services.

☐ **S(1)-4.2 Development of a monitoring environment**
Operators separate systems appropriately to minimize the potential impact of a risk when it occurs, and arrange a monitoring environment to monitor risks that are important to protect assets by means of software.

☐ **S(1)-4.3 Arrangement of a security mechanism**
An appropriate security mechanism is arranged that allows software and systems and services to which the software is applied to protect and monitor the confidentiality and integrity of information assets and data in operating environments or resources such as digital infrastructure.

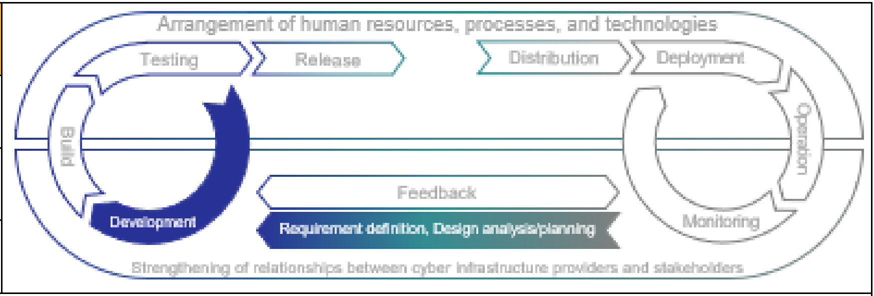
☐ **S(1)-4.4 Monitoring and evaluation**
Operators monitor the operation of mechanisms applied to software that provides important services, periodically conduct security assessments, and integrate them into the risk management framework of the organization.

S(1)-4 requires software operators to monitor whether software-based services operate securely such that information assets and data are protected and maintained through the services. Operations to meet the requirements of S(1)-4 (such as arrangement, support for monitoring, and evaluation of a monitoring system for software used) are generally performed by customers, who are the main entities of software use. However, supposing a case in which specialized knowledge and skills are required to operate a system or service or software that constitutes it, it is assumed that operational support is provided by cyber infrastructure providers based on a contract.

List assets handled by software-based systems/services and assets that constitute a system/service and manage the list to improve the software security at the time of installation and operation and reduce the possibility that software is introduced and operated with vulnerable security settings and exposed to danger.

By introducing and maintaining a secure environment for the operation of software, it is possible to confirm that all components of the software operating environment are appropriately protected from internal and external threats and to prevent the environment or the software that is operated and maintained within it from being compromised. In addition, monitoring the operation status and evaluating the security are expected to be effective for risk management in the operation of important services. As the operating status is to be monitored, it is assumed that the protection mechanism of the software is working effectively to protect information assets and data on resources, and the intended security features of the software are being circumvented or disabled, regardless of whether it is intentional or accidental. To design and implement a security mechanism appropriately and make it possible to monitor its operation, it is desirable to share roles with the developer as necessary.

(2) Life cycle management and assurance of transparency

S (2)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Arrangement of secure software components Verify that commercial, open-source, and other third-party software components procured from outside comply with the defined in-house requirements throughout their life cycles.		

Itemized requirements

☐ **S(2)-1.1 Arrangement of software components**
With respect to commercial, open-source, and other third-party software components procured from outside, adopt those that are highly secure and meet the defined in-house requirements.

☐ **S(2)-1.2 Development and maintenance of software components**
When the software components are not procured from outside, develop highly secure software components in-house in accordance with established in-house security standards and practices, and maintain them.

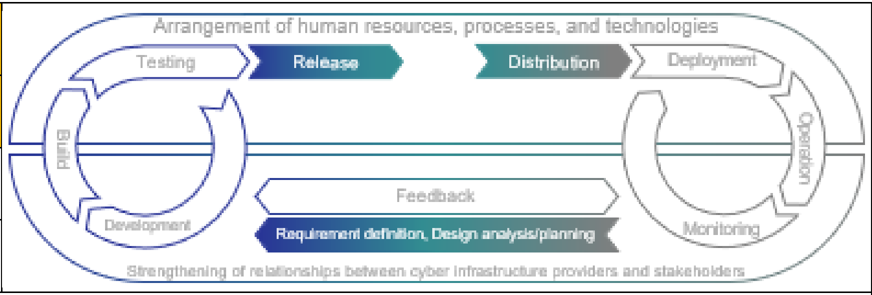
☐ **S(2)-1.3 Risk assessment of software components**
Acquire and analyze information regarding locations from where the respective software components are obtained and assess the risks resulting from the components.

☐ **S(2)-1.4 Confirmation of publicly known vulnerabilities of software components**
Regularly check for publicly known vulnerabilities and periods during which respective software components are supported.

☐ **S(2)-1.5 Updating of software components**
Implement a process to update the respective software components to the new version securely.

S(2)-1 requires software developers to handle third-party software components in compliance with the in-house requirements.

Duplicating functions should be avoided as far as possible and existing secure software components should be used. By reusing software modules and services for which security has been confirmed, and in which update processes for coping with vulnerabilities run appropriately, it is possible to reduce software development costs, accelerate software development, and reduce the possibility of introducing new security vulnerabilities into the software. This is particularly important for software that implements security functions such as cryptographic modules and protocols. Note that when checking for publicly known vulnerabilities, vulnerability information provided by public organizations should be actively utilized.

S (2)-2	Developer	
	Supplier	
	Operator	
	Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Development
Requirement definition, Design analysis/planning
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Secure archiving of release files and data
Archive the necessary files and data to be retained during software release and restrict access to only necessary personnel, tools, and services. Collect, protect, maintain, and share provenance data for all components of the respective releases through the gradual adoption of the SBOM, etc.

Itemized requirements

☐ **S(2)-2.1 Protection of codebases**
To protect codebases in all forms from unauthorized access and tampering, store the codes and configuration information in a repository and implement access control based on the principle of least privilege so that only authorized personnel, tools, and services can access it.

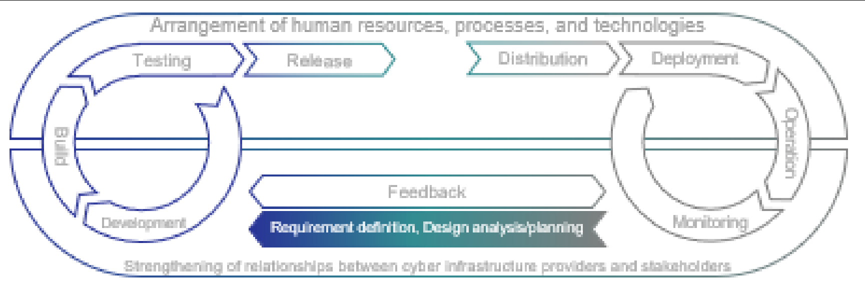
☐ **S(2)-2.2 Archiving of releases**
Archive the respective software releases to protect them so that vulnerabilities identified following release can be analyzed and identified.

☐ **S(2)-2.3 Sharing of release provenance data**
Collect, protect, maintain, and share provenance data for all components of the respective software releases.

S(2)-2 requires software developers and suppliers to archive files and data securely during a software release to protect them.

Protecting all forms of codebases from unauthorized access and tampering helps to prevent invalid changes to codebases that circumvent or disable the intended security properties of software, regardless of whether they are intentional or accidental. Codes that are not made public help to prevent software theft, making it more difficult for attackers to identify software vulnerabilities.

Archiving software releases to protect them can assist in identifying, analyzing, and removing vulnerabilities identified in the software after it is released. Note that to securely archive necessary files and support data that should be retained during a software release (e.g., integrity verification information, provenance data) and make them shareable with stakeholders requires tasks related to the generation, maintenance, and sharing of component lists using SBOM.

S (2)-3	Developer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Establishment of security requirements among stakeholders
Establish security requirements for the parties involved to agree upon and include them in contracts or policies to be shared.

Itemized requirements

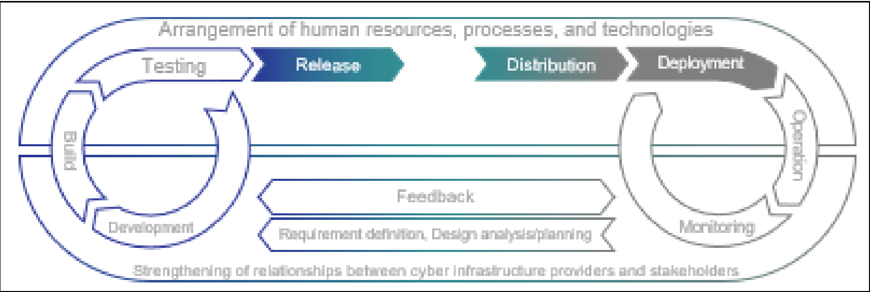
☐ **S(2)-3.1 Agreement on security requirements**
Include explicit security requirements in contracts or policies to be shared with third parties that provide software (including commercial software components for use in in-house software) or services.

☐ **S(2)-3.2 Responses to supply chain security requirements**
Respond to supply chain security requirements equivalent to those adopted by the organization that receives or acquires third-party software or services that it provides.

☐ **S(2)-3.3 Establishment of a response process for risks that do not meet security requirements**
Arrange a process to respond to risks in the case in which there are security requirements that third-party software or services to be received or acquired do not meet.

S(2)-3 requires software developers, suppliers, and operators to establish security requirements to be shared among the parties involved.

By explicitly defining software development and operation security requirements (including those in supply chains. E.g., management of development environments, management of subcontractors, inspection standards for procured items) in contracts or policies to be shared with third parties and making them always known to the parties involved, it is possible to consider security requirements (including those in supply chains) throughout the SDLC. Furthermore, by sharing requirements completely and reliably, the duplication of effort can be minimized. Note that operations to meet the requirement of S(2)-3 (agreement on security requirements for IT products and services necessary for the operation of software and support for the arrangement of related risk response processes) are generally carried out by the customer, who is the main entity of software use. However, supposing a case in which specialized knowledge and skills that third parties possess are required to operate a system or service or software that constitutes it, it is assumed that the operational support is provided by cyber infrastructure providers based on a contract with the third parties or a policy shared with the third parties.

S (2)-4	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Appropriate information provision to users
Ensure that software users can apply guidance that facilitates secure use throughout the entire software life cycle—from introduction and installation to operation and termination of use.

Itemized requirements

☐ **S(2)-4.1 Secure introduction, configuration, operation, modification, disposal, and termination**
Ensure that software users can continuously use information for securely introducing, configuring, and operating software, as well as information related to the impact of changes, disposal, termination of provision, and termination of use.

☐ **S(2)-4.2 Provision of integrity verification information**
Ensure that software users can continuously use information that is necessary for verifying the integrity and completeness of the software.

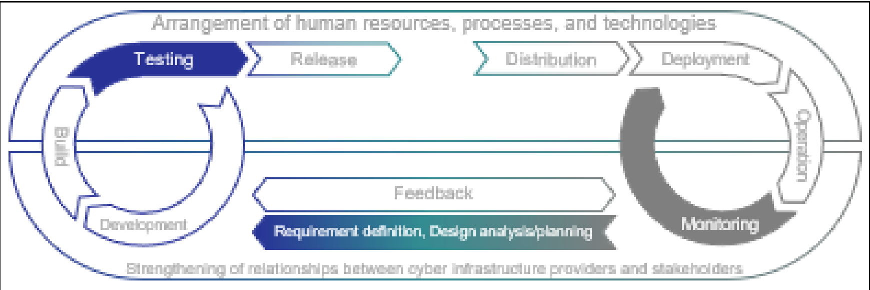
S(2)-4 requires software developers and suppliers to provide users with information to use software securely.

Providing information for securely introducing, configuring, and operating software improves the security of the software at the time of installation and reduces the possibility of exposure to risks—for example, when the software is introduced with weak security settings and operated in an insecure manner. In addition, making information available on the end of sale (EOS)/end of life (EOL) (end of maintenance and support) of a product/service[14], as well as on the impact of change, disposal, termination of provision, and termination of use, helps software users to manage assets and operate the software securely. Even after the software is supplied, developers must continue to provide such information to users.

In addition, secure default settings for the software (or, if applicable, a default configuration or a group of interrelated default settings) should be implemented and information regarding the respective settings should be provided to software administrators. Providing a mechanism for verifying the integrity of software releases helps software users to ensure that the software that they acquire is genuine and has not been tampered with.

[14] In the guidelines, EOS refers to the end of sale of a product/service, and EOL refers to the end of life of a product/service. Other terms similar to EOL, such as EOSL (end of service life), EOS (end of support), EOS (end of service), and EOE (end of engineering), may be used.

(3) Prompt responses to remaining vulnerabilities

S (3)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Development Feedback Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Continuous vulnerability investigation Establish a policy for disclosure and remediation of software vulnerabilities; define roles, responsibilities, and processes required for the policy and implement them.		

Itemized requirements

☐ **S(3)-1.1 Establishment of a vulnerability response system**
Establish a policy for the disclosure and remediation of vulnerabilities of software products, establish a system for responses to vulnerabilities (including responses to incidents) to support the policy, and define necessary roles, responsibilities, and processes.

☐ **S(3)-1.2 Communication plan**
Establish a communication plan for all stakeholders.

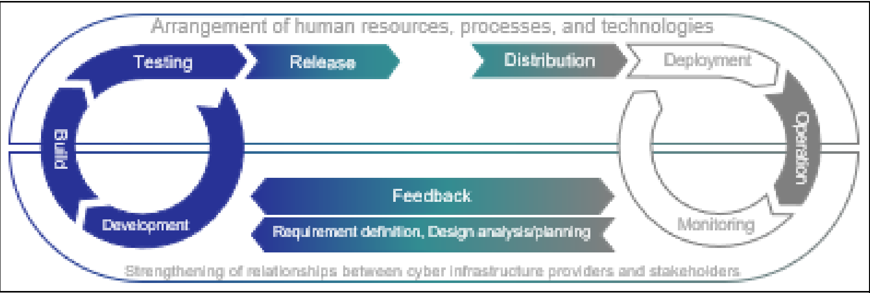
☐ **S(3)-1.3 Vulnerability information collection**
Collect new information regarding vulnerabilities through searches of public information, notifications from software users, the acquisition of external threat information, reviews of system configuration data, and other methods.

☐ **S(3)-1.4 Identification of undetected vulnerabilities**
Conduct software code review, analysis, and testing on an ongoing or regular basis to identify undetected vulnerabilities (including improper settings) to be solved.

S(3)-1 requires software developers and operators to establish a system related to vulnerability responses, including software incident responses, and to conduct ongoing vulnerability investigations based on a policy related to vulnerability disclosure and correction. In particular, developers must continually address vulnerabilities in the software that they have designed and developed.

Note that, with regard to operation, efforts to meet the requirement of S(3)-1 (such as support for responding to incidents and for collecting information on vulnerabilities in the software used) are generally performed by customers who are the main entities of software use. However, supposing a case in which specialized knowledge and skills are required to operate a system or service or software that constitutes it, it is assumed that operational support is provided by cyber infrastructure providers based on a contract.

Continuously identifying and verifying vulnerabilities makes it possible to identify vulnerabilities more quickly and take measures, such as promptly correcting them according to the risk, which ultimately contributes to reducing opportunities for attackers to launch attacks. Software developers establish policies for disclosing and correcting vulnerabilities and implement the roles, responsibilities, and processes necessary to promote responses based on these policies. Software operators provide software developers with information regarding vulnerabilities that may be latent in the software and its third-party components.

S (3)-2	Developer	
	Supplier	
	Operator	
	Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Development
Requirement definition, Design analysis/planning
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Responses to detected vulnerabilities
Regularly create a plan to respond to risks of vulnerabilities remaining in released software and implement it.

Itemized requirements

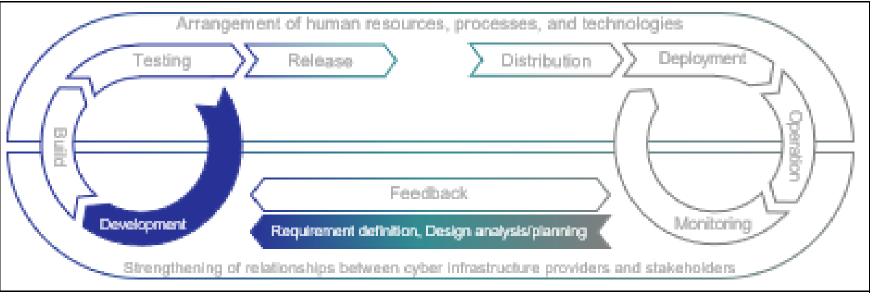
☐ **S(3)-2.1 Vulnerability analysis**
Developers collect information necessary to understand the risks associated with the impact of each remaining vulnerability and analyze each vulnerability to plan remediations or other responses to risks.

☐ **S(3)-2.2 Risk responses to vulnerabilities**
Developers create a plan for risk responses for each vulnerability and implement it.

☐ **S(3)-2.3 Security recommendations**
Developers prepare security recommendations, provide the information to the supplier of the released software, and create a report as specified by the relevant systems. In addition, operators implement deployment in accordance with security recommendations.

S(3)-2 requires software developers to evaluate, prioritize, and correct vulnerabilities.

By analyzing each vulnerability, collecting sufficient information regarding the risk, planning correction thereof or other risk responses, and implementing software corrections, vulnerabilities can be corrected in response to risk and help to reduce opportunities for attackers to launch attacks. In particular, when information regarding exploited vulnerabilities is provided by public institutions, it is required to respond appropriately and proactively, including patch development. In addition, providing security recommendations and patches to suppliers and applying them will lead to the maintenance of secure software operations.

S (3)-3	**Developer**	
	Supplier	
	Operator	
	Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Development
Requirement definition, Design analysis/planning
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Application of results of countermeasures to in-house process improvements

Based on the analyses of vulnerabilities (root causes), review development and operation processes so that the root causes of problems identified in the software do not recur or the possibility of their recurrence is reduced.

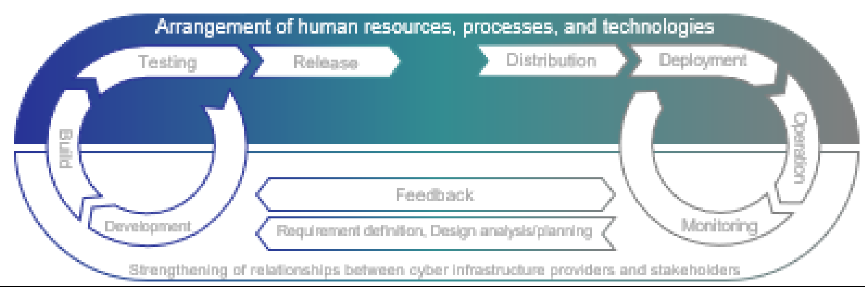
Itemized requirements

- ☐ **S(3)-3.1 Identification of root causes**
  Analyze an identified vulnerability to determine its root causes and proactively take countermeasures.
- ☐ **S(3)-3.2 Process improvement**
  Review development and operation processes for the entire software life cycle and revise them as necessary to prevent root causes from recurring or reduce the possibility of their recurrence through software updates or new software creation.

S(3)-3 requires software developers and operators to identify root causes by analyzing vulnerabilities and implementing countermeasures. Note that operations to meet the requirement of S(3)-3 (such as improvement of the software use process and support for root cause analysis) are generally performed by customers, who are the main entities of software use. However, supposing a case in which specialized knowledge and skills are required to operate a system or service or software that constitutes it, it is assumed that operational support is provided by cyber infrastructure providers based on a contract.

By analyzing the identified vulnerabilities, identifying their root causes, and taking countermeasures, the frequency of vulnerabilities that will occur in the future can be reduced. In addition, by reviewing SDLC processes and updating them such that root causes do not recur (or the possibility that root causes are lowered) in software updates and newly created software, the possibility that root causes will recur can be prevented or reduced, thereby contributing to a reduction in the frequency of vulnerabilities.

(4) Arrangement of human resources, processes, and technologies

S (4)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

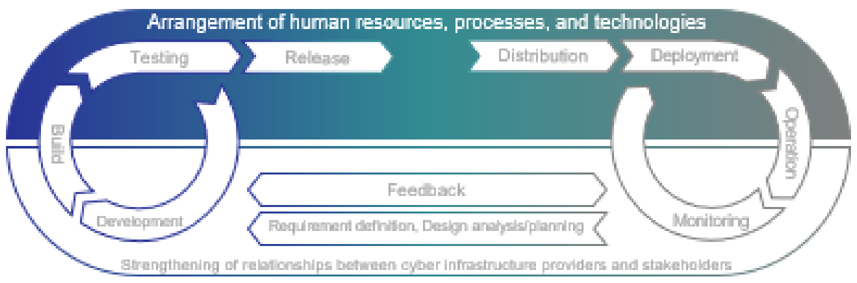
Human resources: Commitment from management and arrangement of personnel
Define roles and responsibilities covering the entire software life cycle. Make management's commitment to secure development known, secure personnel for security measures, provide training to all personnel related to secure development and operation according to their levels of proficiency and role, and review it regularly.

Itemized requirements

- ☐ **S(4)-1.1 Definition of roles and responsibilities**
  Define roles and responsibilities covering the entire software development life cycle.
- ☐ **S(4)-1.2 Management's commitment**
  Make management's commitment to secure development known to all personnel, and educate them on the importance of secure development and operation to the organization.
- ☐ **S(4)-1.3 Agreement on roles and responsibilities**
  Confirm that all personnel are aware of and agree to their roles and responsibilities.
- ☐ **S(4)-1.4 Training for each role**
  Create a training plan for each role and implement it so that all personnel can be trained according to their level of proficiency and role.
- ☐ **S(4)-1.5 Review of roles and training**
  Review roles and training regularly.

S(4)-1 requires software developers, suppliers, and operators to clarify the roles and responsibilities of the personnel involved in the SDLC and provide appropriate training according to the role. Note that operations to meet the requirement of S(4)-1 (such as training for operators' roles) are generally performed by customers who are the main entities of software use. However, supposing a case in which specialized knowledge and skills are required to operate a system or service or software that constitutes it, it is assumed that cyber infrastructure providers provide operational support based on a contract.

By clearly determining roles and responsibilities in software development and providing training according to these roles, everyone engaged with the SDLC, both inside and outside an organization, will be prepared to fulfill the roles and responsibilities related to the SDLC throughout the SDLC. In addition, roles and responsibilities should regularly be reviewed and training reviewed and updated according to the proficiency and role of the personnel to maintain security response capabilities over the entire SDLC.

S (4)-2	Developer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Process: Establishment of development policy and compliance with laws and regulations
Comply with laws and regulations, document and maintain a security policy for in-house development infrastructures and processes, and secure necessary budgets for security securement.

Itemized requirements

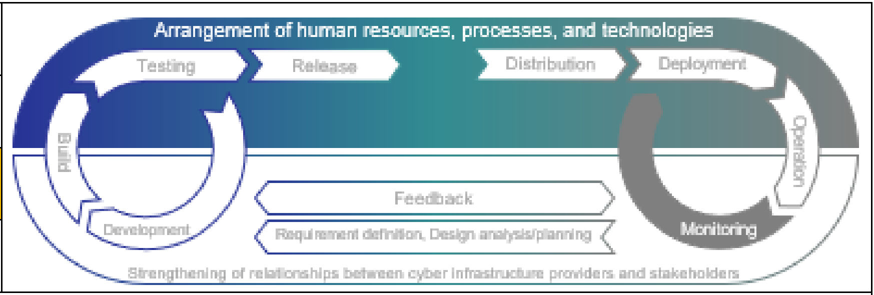
☐ **S(4)-2.1 Definition of a software development policy**
Identify all security requirements for software development infrastructures and processes (including requirements related to EOL), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.

☐ **S(4)-2.2 Definition and maintenance of a software security policy**
Define a policy that specifies all security requirements that must be met by the software developed by an organization, and maintain the requirements throughout the SDLC.

☐ **S(4)-2.3 Sharing of cost recognition and budgeting**
Secure necessary budgets to ensure security based on a policy.

S(4)-2 requires software developers to establish a security policy for in-house development infrastructures and processes and to maintain it throughout the SDLC in compliance with laws and regulations (including budget securement).

Security requirements for the software development infrastructure and processes, as well as security requirements that the software must meet, should be identified. By defining a policy to maintain the requirements throughout the SDLC and making software development security requirements (including requirements related to EOL) identifiable at any time, it is possible to consider them throughout the SDLC. In addition, sharing software development requirements helps to minimize the duplication of effort. Furthermore, when considering budgets for ensuring security, a policy provides a basis for stakeholders to share their understanding.

S (4)-3	Developer	
	Supplier	
	Operator	
	Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Development
Operation
Monitoring
Feedback
Requirement definition, Design analysis/planning
Strengthening of relationships between cyber infrastructure providers and stakeholders

Process: Establishment of an operation policy and compliance with laws and regulations
Comply with laws and regulations, and document and maintain all security policies for service operation infrastructures and processes to which the software is applied.

Itemized requirements

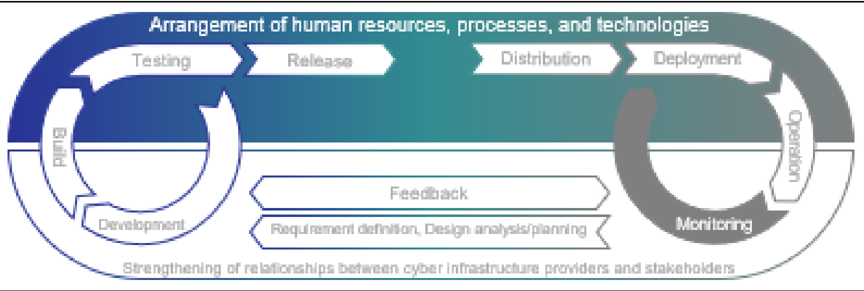
☐ **S(4)-3.1 Definition of a software service operation policy**
Identify all security requirements for service operation infrastructures and processes to which the software is applied (including requirements related to EOS and disposal), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.

☐ **S(4)-3.2 Definition and maintenance of a service security policy**
Define a policy that specifies all security requirements that services to which the software is applied must meet, and maintain the requirements throughout the SDLC.

☐ **S(4)-3.3 Audit based on an operation policy**
Confirm through an audit that the protection of service operation infrastructures and processes and security requirements for service are maintained throughout the SDLC in accordance with policy-based governance.

S(4)-3 requires software operators to establish a security policy for software operation infrastructures and processes and maintain it throughout the SDLC in compliance with laws and regulations (including budget securement). Note that operations to meet the requirement of S(4)-3 (such as definition, maintenance, and policy-based audit support) are generally performed by the customer, who is the main entity of software use. However, supposing a case in which specialized knowledge and skills are required to operate the system/service or the software that constitutes it, it is assumed that operational support is provided by cyber infrastructure providers based on a contract.

Security requirements for the operation of services to which the software is applied and security requirements that a service to which the software is applied must meet should be identified. By defining a policy to maintain the requirements throughout the SDLC and making software operation security requirements (including requirements related to EOL) identifiable at any time, it is possible to provide consideration throughout the SDLC and make requirements related to software operation shareable, minimizing the duplication of effort. In addition, through audits, the governance status based on an operation policy can be identified and maintenance of security requirements can be implemented over the long term throughout the SDLC.

S (4)-4	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Process: Establishment of development and operational standards
Define security verification criteria related to software development and operation, collect information necessary to support the criteria, and implement processes and mechanisms for conformance. Track the status of conformance throughout the entire life cycle.

Itemized requirements

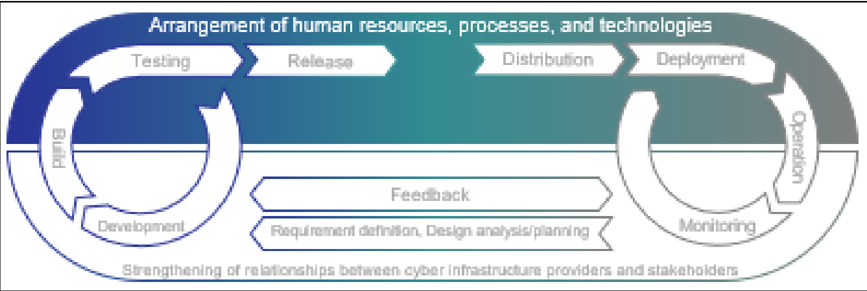
☐ **S(4)-4.1 Definition and tracking of security verification criteria**
Define software security verification criteria and track the entire SDLC.

☐ **S(4)-4.2 Support for decision-making based on security verification criteria**
Implement processes and mechanisms for collecting and protecting information necessary to support decision-making based on security verification criteria.

☐ **S(4)-4.3 Audit based on security verification criteria**
Track the entire SDLC and verify through audits that the intended effects are achieved with governance to ensure conformance to security verification criteria.

S(4)-4 requires software developers and operators to collect information based on the criteria for verifying software security and track conformance to the criteria. Note that operations to meet the requirement of S(4)-4 (such as decision-making support and audit support based on security confirmation criteria for operation) are generally performed by the customer, who is the main entity of software use. However, supposing a case in which specialized knowledge and skills are required to operate the system/service or the software that constitutes it, it is assumed that operational support is provided by cyber infrastructure providers based on a contract.

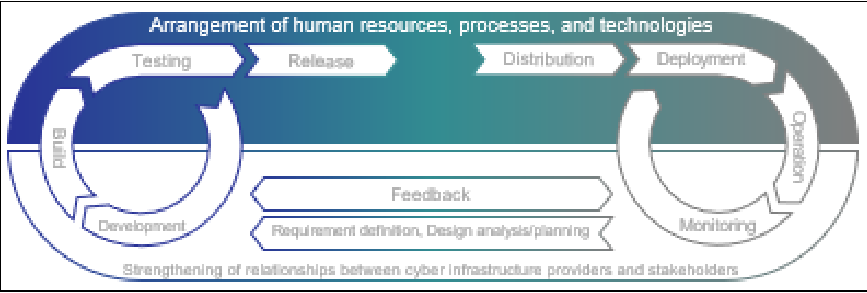
By defining the criteria for confirming software security and tracking the status of security implementation throughout the SDLC, it is possible to use them as a standard for checking the security of the software to be developed and maintained. Meeting the standard (assurance) helps to ensure that the software continues to meet the organizational expectations obtained from the SDLC and ensures its security. In addition, through audits, the governance status for conformance and compliance with confirmation criteria can be assessed and security levels throughout the SDLC can be maintained over the long term.

S (4)-5	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Technology: Arrangement of secure development tools Analyze risks throughout the SDLC and implement security measures in development tools.		

Itemized requirements
☐ **S(4)-5.1 Designation of tools and toolchains** Identify tools that are effective in mitigating specific risks and determine means of integrating toolchain components mutually.
☐ **S(4)-5.2 Deployment, operation, and maintenance of tools and toolchains** Deploy, operate, and maintain tools and toolchains in accordance with security practices.
☐ **S(4)-5.3 Tool configuration and evidence generation** Configure tools to generate evidence regarding support for secure software development practices defined in-house.

S(4)-5 requires software developers to implement security measures in software development tools.

The use of toolchains to support software development makes it possible to promote automation and reduce human effort. In addition, by providing a method to document and demonstrate the utilization of these measures, it is possible to improve the accuracy, repeatability, ease of use, and comprehensiveness (overall connectedness of development) of the security measures throughout the SDLC. In addition, toolchains and tools can be used at various organizational levels, such as organization-wide or project-specific levels, and some can be used to generate evidence of the software development implementation status automatically, contributing to the automation of specific sessions of the SDLC and feedback effects for process reviews.

S (4)-6	Developer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Technology: Arrangement of secure development environments
Analyze risks throughout the SDLC, and protect and strengthen development-related environments.

Itemized requirements

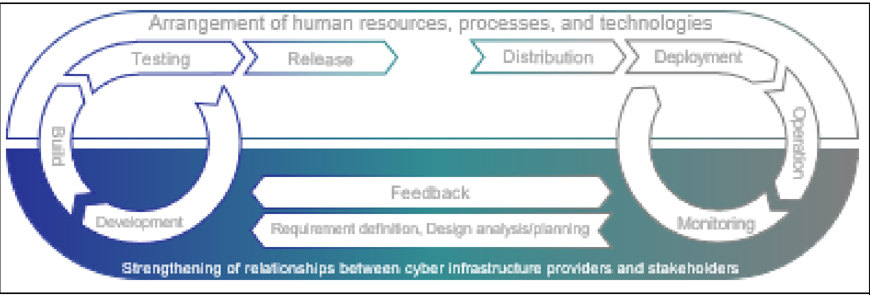
☐ **S(4)-6.1 Isolation and protection of environments**
Isolate and protect the respective environments related to software development.

☐ **S(4)-6.2 Protection of development endpoints**
Protect and strengthen endpoints designed for respective developers to perform development-related tasks using a risk-based approach.

S(4)-6 requires software developers to establish secure development environments.

By implementing secure development environments for software development and maintaining the protected state of endpoints designed for development (software architects, developers, testers, etc.) to perform development-related tasks using a risk-based approach, it is possible to ensure that all components of software development environments are adequately protected from internal and external threats. This helps to prevent development environments and the software developed and maintained therein from being compromised.

(5) Strengthening of relationships between cyber infrastructure provider and stakeholders

S (5)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Organizational system for information sharing Establish an organizational structure for information sharing between private companies, relevant authorities, and specialized organizations to improve the security of software products and services.		

Itemized requirements

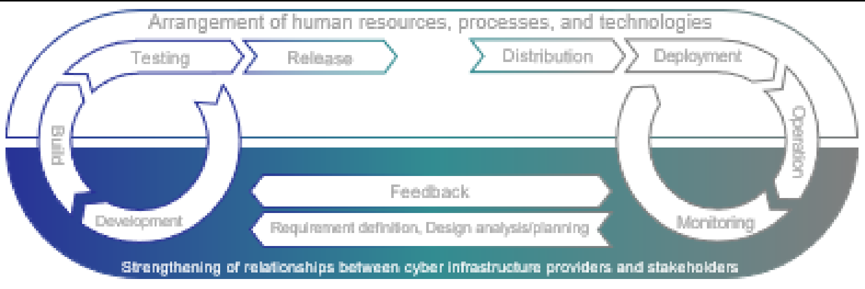
☐ **S(5)-1.1 Establishment of an organizational system for information sharing**
Establish an organizational structure for information sharing between private companies, relevant authorities, and specialized organizations to improve the security of software products and services.

☐ **S(5)-1.2 Provision of important security-related information**
Select and identify essential and important security-related information that is specific to the industry and provide it to partners in the supply chain.

☐ **S(5)-1.3 Use of vulnerability information notification services**
Use vulnerability information notification services to share vulnerability information efficiently.

S(5)-1 requires software developers, suppliers, and operators to establish an organizational system for information sharing aimed at improving software security.

Information related to software security includes information regarding vulnerabilities (the impact of attacks and countermeasures, damage examples, etc.), legal requirements, and industry best practices. Arranging an organizational system to obtain, provide, and share such information and strengthen relationships for information sharing with stakeholders will help to improve software security continuously.

S (5)-2	Developer Supplier Operator Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Development
Requirement definition, Design analysis/planning
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Strengthening of cooperation systems
To improve the security of software products and services, make use of systems and frameworks for cooperation with private companies, relevant authorities, and specialized organizations.

Itemized requirements

☐ **S(5)-2.1 Utilization of cooperation systems**
To improve the security of software products and services, make use of communities and cooperation systems aimed at improving software security, in which external businesses, customers, and specialized organizations participate.

☐ **S(5)-2.2 Contribution to cooperation systems**
When participating in a community or cooperation system, actively participate in activities to contribute to the cooperation system.

S(5)-2 requires software developers, suppliers, and operators to make use of cooperation systems aimed at improving software security.

Participating in communities and cooperation systems for improving software security and contributing to their activities deepens the mutual understanding of responsibilities and roles to ensure, maintain, and improve security and increases the effectiveness and efficiency of actions to improve security, thereby ensuring software security and improving resilience.

(6) Risk management by customers, and procurement and operation of secure software

S (6)-1	Developer	Supplier	Operator	Customer
Risk management under the leadership of the customer's management Integrate risk management that is implemented in cooperation with cyber infrastructure providers based on the leadership of the customer's management.				

Itemized requirements
☐ **S(6)-1.1 Risk management** Implement risk management in which the customer's independent and proactive efforts are integrated with efforts based on a contract with cyber infrastructure providers.
☐ **S(6)-1.2 Resource arrangement** Allocate and develop resources to respond proactively to known vulnerabilities and implement mitigation measures (including SBOM utilization).
☐ **S(6)-1.3 Utilization of collaborative systems** Proactively participate in and utilize communities and collaborative systems aimed at improving software security.

S(6)-1 requires customers to implement their own risk management in cooperation with cyber infrastructure providers based on the leadership of management.

Promoting risk management related to software security through the leadership of the customer's management encourages customers to ensure software security and improve resilience. To achieve this, it is necessary to clarify the responsibilities and roles of risk response with cyber infrastructure providers, who are trading partners, and manage risks in an integrated manner according to the procedures agreed upon by the contract. In addition, it is necessary to prepare the resources required to respond to known vulnerabilities and mitigation measures based on software usage life cycles. Participating in communities and cooperation systems for improving software security and contributing to their activities deepens the mutual understanding of responsibilities and roles to ensure, maintain, and improve security, and increases the effectiveness and efficiency of actions to improve security, thereby ensuring software security and improving resilience.

S (6)-2	Developer	Supplier	Operator	Customer
Software procurement/operation under the leadership of the customer's management Procure and operate software securely under the leadership of the customer's management.				

Itemized requirements

☐ **S(6)-2.1 Definition of security requirements**
Define security requirements for incorporating security functions into software design plans and present them to cyber infrastructure providers before procuring and deploying software.

☐ **S(6)-2.2 Disclosure of security practice requirements**
Disclose security practice requirements for cyber infrastructure providers before procuring and deploying software.

☐ **S(6)-2.3 Decision-making based on risk assessment**
When procuring and introducing software, make decisions based on risk assessment.

☐ **S(6)-2.4 Budget securement**
Continuously secure budgets related to introduction, operation, migration, disposal, risk response, and related contracts, considering software life cycles.

S(6)-2 requires customers to procure and operate software securely under the leadership of their management.

When procuring and operating software under the leadership of the customer's management, these steps must be followed to ensure software security, and improved software resilience: indicating defined security requirements and security practice requirements for cyber infrastructure providers to businesses that may be selected as contractors; making decisions on procuring and introducing software based on proper risk assessment; and securing the necessary budgets for the respective phases of deployment, operation, migration, and disposal, considering software life cycles, as well as proper and continuous risk response. The security practices required of cyber infrastructure providers (including supply chain security measures) should be specified based on the characteristics of the software to be procured and deployed and an acceptable judgment of risk measures should be provided.

4. Utilization of requirements

4.1. Requirement packaging of requirements

Requirements that cyber infrastructure providers and customers (users of software products and services, including government agencies and critical infrastructure operators) must address to fulfill their responsibilities to improve resilience regarding software cybersecurity are classified according to the purpose and goal of the requirement into the following two categories, and can be used as a requirements package (itemized requirements):

- **Minimum requirement package**
  A group of requirements (itemized requirements) that all cyber infrastructure providers and customers must implement at a minimum. These are limited to secure procurement, responding to vulnerabilities before and during software supply, and sharing of minimum necessary information.

- **Standard requirement package**
  A group of requirements (itemized requirements) that must be implemented as a standard. These include the establishment of a secure development and risk response systems, and cooperation between stakeholders. From the perspective of information handled by the software, to ensure prompt maintenance of the mechanism for protecting it and rapid responsiveness to vulnerabilities and reliability issues, particularly when a lack of these is considered a risk, the standard requirement package should be applied.

The relationships between the requirements and requirement packages are shown in Table 5 (for cyber infrastructure providers) and Table 6 (for customers).

Table 5 Relationship between the requirements for cyber infrastructure providers and requirements packages

Requirements for cyber infrastructure providers (itemized requirements)	Developer	Supplier	Operator	Minimum requirement package	Standard requirement package
S(1)-1.1 Risk-based security requirements definition	√			○	○
S(1)-1.2 Design review	√			○	○
S(1)-1.3 Risk response records	√				○
S(1)-1.4 Periodic risk-based review	√				○
S(1)-2.1 Definition of secure development process	√			○	○
S(1)-2.2 Secure build	√			○	○
S(1)-2.3 Verification and feedback	√			○	○
S(1)-2.4 Codebases	√			○	○
S(1)-3.1 Test planning	√			○	○
S(1)-3.2 Test method	√			○	○
S(1)-3.3 Test implementation	√			○	○
S(1)-3.4 Response to problems	√			○	○
S(1)-4.1 Asset management			√	○	○
S(1)-4.2 Development of a monitoring environment			√		○

Requirements for cyber infrastructure providers (itemized requirements)	Developer	Supplier	Operator	Minimum requirement package	Standard requirement package
S(1)-4.3 Arrangement of a security mechanism	✓		✓	▨	○
S(1)-4.4 Monitoring and evaluation			✓	○	○
S(2)-1.1 Arrangement of software components	✓			○	○
S(2)-1.2 Development and maintenance of software components	✓			○	○
S(2)-1.3 Risk assessment of software components	✓			○	○
S(2)-1.4 Confirmation of publicly known vulnerabilities of software components	✓			○	○
S(2)-1.5 Updating of software components	✓			○	○
S(2)-2.1 Protection of codebases	✓	✓		○	○
S(2)-2.2 Archiving of releases	✓	✓		○	○
S(2)-2.3 Sharing of release provenance data	✓	✓	✓	○	○
S(2)-3.1 Agreement on security requirements	✓	✓	✓	○	○
S(2)-3.2 Response to supply chain security requirements	✓	✓		▨	○
S(2)-3.3 Establishment of a response process for risks that do not meet security requirements	✓	✓	✓	▨	○
S(2)-4.1 Secure introduction, configuration, operation, modification, disposal, and termination	✓	✓		○	○
S(2)-4.2 Provision of integrity verification information	✓	✓		○	○
S(3)-1.1 Establishment of a vulnerability response system	✓		✓	○	○
S(3)-1.2 Communication plan	✓		✓	○	○
S(3)-1.3 Vulnerability information collection	✓		✓	○	○
S(3)-1.4 Identification of undetected vulnerabilities	✓		✓	○	○
S(3)-2.1 Vulnerability analysis	✓			○	○
S(3)-2.2 Risk response to vulnerabilities	✓			○	○
S(3)-2.3 Security recommendations	✓	✓	✓	○	○
S(3)-3.1 Identification of root causes	✓		✓	▨	○
S(3)-3.2 Process improvement	✓		✓	▨	○
S(4)-1.1 Definition of roles and responsibilities	✓	✓	✓	▨	○
S(4)-1.2 Management's commitment	✓	✓	✓	○	○
S(4)-1.3 Agreement on roles and responsibilities	✓	✓	✓	▨	○
S(4)-1.4 Training for each role	✓	✓	✓	▨	○
S(4)-1.5 Review of roles and training	✓	✓	✓	▨	○
S(4)-2.1 Definition of a software development policy	✓			○	○
S(4)-2.2 Definition and maintenance of a software security policy	✓			○	○
S(4)-2.3 Sharing of cost recognition and budgeting	✓			○	○
S(4)-3.1 Definition of a software service operation policy			✓	▨	○
S(4)-3.2 Definition and maintenance of a service security policy			✓	▨	○
S(4)-3.3 Audit based on an operation policy			✓	▨	○

Requirements for cyber infrastructure providers (itemized requirements)	Developer	Supplier	Operator	Minimum requirement package	Standard requirement package
S(4)-4.1 Definition and tracking of security verification criteria	✓		✓	○	○
S(4)-4.2 Support for decision-making based on security verification criteria	✓		✓	○	○
S(4)-4.3 Audit based on security verification criteria	✓		✓		○
S(4)-5.1 Designation of tools and toolchains	✓			○	○
S(4)-5.2 Deployment, operation, and maintenance of tools and toolchains	✓			○	○
S(4)-5.3 Tool configuration and evidence generation	✓			○	○
S(4)-6.1 Isolation and protection of environments	✓			○	○
S(4)-6.2 Protection of development endpoints	✓			○	○
S(5)-1.1 Establishment of an organizational system for information sharing	✓	✓	✓		○
S(5)-1.2 Provision of important security-related information	✓	✓	✓	○	○
S(5)-1.3 Use of vulnerability information notification services	✓	✓	✓	○	○
S(5)-2.1 Utilization of cooperation systems	✓	✓	✓		○
S(5)-2.2 Contribution to cooperation systems	✓	✓	✓		○

Table 6 **Relationship between the requirements for cyber infrastructure providers and requirements packages**

Requirements for customers (itemized requirement)	Minimum requirement package	Standard requirement package
S(6)-1.1 Risk management	○	○
S(6)-1.2 Resource arrangement	○	○
S(6)-1.3 Utilization of collaborative systems		○
S(6)-2.1 Definition of security requirements	○	○
S(6)-2.2 Disclosure of security practice requirements	○	○
S(6)-2.3 Decision-making based on risk assessment	○	○
S(6)-2.4 Budget securement	○	○

4.2. Points to note regarding the application of requirements according to the division of roles

The basic procedure to set the requirements to be responded to based on the relationship between the division of roles and the requirements for cyber infrastructure providers and the points to note for appropriately applying the requirements to be responded to are described below.

- Basic procedure to set requirements according to the division of roles for cyber infrastructure providers

Cyber infrastructure providers are expected to clarify the scopes of their organizational roles for the intended software (whether or not they have the roles of developer, supplier, and operator), determine the degrees of achievement of the necessary requirements (standard or minimum for requirement packages), and consider and implement cybersecurity measures that meet the itemized requirements according to their roles. In general, it is desirable to set identical degrees of achievement (standard or minimum requirement packages) for the entire supply chain—including suppliers of software components and software development contractors (up to the end of development outsourcing)—thereby setting degrees of achievement of the demanded requirements such that they are consistent as role scopes of cyber infrastructure providers to allow customers to meet the degrees of achievement themselves. When assuming a role of the main entity, it is necessary to meet all requirements of a set degree of achievement (standard or minimum requirement package). Even if a supporting role is assumed, it is desirable to meet degrees of achievement and requirements equivalent to the main entity of the role; however, it is acceptable to provide them as responses limited to the requirements assigned to the main entity depending on the responsibility in the scopes to be supported.

- Establishment of a CSIRT by the customer

When the system development is completed and the system enters the operational phase through customer acceptance, the customer may establish a CSIRT to respond to system incidents, take the lead in responding to software vulnerabilities, and outsource the actual work of responding to vulnerabilities in some software to a cyber infrastructure provider. In such a case, it is assumed that the customer's operation department carries out requirement S(3) as the operator, and the outsourced cyber infrastructure provider performs the operation of requirement S(3), which is intended to respond to vulnerabilities in some software, as the operator.

- Application of a code generation tool by the customer

In development using a no-code platform, in which a customer generates code using a development code generation tool provided by a cyber infrastructure provider, there is a risk that programs that cyber infrastructure providers do not anticipate will be generated. In the case of software generated with such a procedure, the code generation tool is regarded as a tool to be used by the developer to perform part of the development activity and by the customer (a development department of the customer, etc.) for testing to ensure proper operation of the software based on customer specifications. In this case, it is desirable to organize the division of roles individually; for example, the cyber infrastructure provider assumes the roles of the developer and supplier of the code generation tool (or the no-code platform, which includes it) as a software product, and the customer itself (a development department of the customer, etc.) assumes the role of the developer of the software generated by applying the tool.

5. Reference information

5.1. Requirements checklist

Refer to the attached "Requirements Checklist" and "Requirements Checklist (Roles/Phases)", which provide information regarding requirements in the form of a table.

5.2. Examples of relationships between security incidents and requirements

For cases of security incidents with a major impact on society, how the requirements organized in the Guidelines reduce risks and their correspondence relationships are described below as reference information.

■ Apache Log4J vulnerability

Apache Log4J is a logging library used worldwide. In 2021, a serious vulnerability that allowed attackers to execute arbitrary code remotely was discovered in it and exploited. As it is incorporated into various types of software in multi-layered software supply chains, it causes vulnerabilities that are not easily found, tracked, and fixed. In some cases, it allows vulnerabilities to remain for a long time.

In this case, it is possible to reduce risks by collecting vulnerability information and formulating a response, as specified in requirement S(3), and by understanding vulnerability information by establishing an information collection system, as specified in requirement S(5). In addition, even in cases in which software development begins after the vulnerability information is made public, it is possible to eliminate the use of the software in which vulnerabilities remain unsolved and adopt an appropriate software, as specified in requirement S(2).

■ Incident in Software Vendor A

This is a case in which a legitimate software update was tampered with, affecting the entire organization using the software. A software update was tampered with by intrusion into the software development company's development and operation environment, and the security of the development and operation environment from upstream to downstream of its software supply chain was not sufficiently ensured.

Here, it is possible to reduce risks by making it difficult for attackers to intrude by creating a secure development and operation environment with prompt maintenance, as per requirements S(1) and S(4).

■ Encryption and leakage incident of patient information held by Hospital B

This is a case where an attacker exploited a vulnerability in a VPN device to intrude into the hospital's network, encrypting and leaking patient information maintained by the hospital, consequently causing problems in medical treatment operations. This occurred because vulnerability of the VPN device was neglected.

Here, it is possible to reduce risks by collecting vulnerability information and formulating a response, as specified in requirement S(3), and by understanding vulnerability information by establishing an information collection system, as specified in requirement S(5). In addition, the customer side can reduce risks by cooperating with businesses through the procurement and operation of secure software, as stated in requirement S(6).

5.3. Correspondence relationships between threats in a system life cycle and requirements

The main correspondence relationships between threats and requirements are described in Table 7 as reference information. These outline the importance of the requirements organized in the Guidelines in handling threats in a system life cycle.

Table 7 Correspondence relationships between system life cycle, threats, and requirements

System life cycle	Outline of the threat	Reason for requirements against the threat
Analysis/Planning	• **Insufficient current situation analysis** A system/service in which current vulnerabilities and security are not considered is built without sufficient risk analysis of the current system/service between the customer and developer/supplier.	Businesses appropriately define security requirements through risk analysis as per S(1)-1 and confirm the results of the analysis. Customers make decisions based on risk assessment, as per S(6)-2. With these measures, an appropriate analysis of the current situation is conducted.
Requirement definition	• **Disagreement on requirements** Without sufficient agreement on security requirements between the customer and business, unintended security requirements are defined. This leads to misunderstanding or a lack of security requirements.	Businesses define appropriate security requirements as per S(1)-1 and agree on security requirements between businesses as per S(2)-3, dealing with risks that do not meet requirements. Customers proactively engage in the definition of security requirements as per S(6)-2. With these measures, the customer and business agree on appropriate security requirements.
Design - testing	• **Misunderstanding of requirements/Improper implementation** Security requirements are not fully understood or not properly implemented from the perspective of software security quality.	Businesses maintain appropriate risk response by recording risk responses and continuously reviewing risk response measures as per S(1)-1. In addition, as per S(1)-2, appropriate implementation is conducted by making use of a secure development process. With these measures, requirements are implemented properly.
	• **Intentional code manipulation** By exploiting an insecure development environment, an attacker intentionally injects malicious code or components such as a backdoor that enables future unauthorized access. Alternatively, an attacker is allowed to steal confidential information such as source code.	Businesses implement access control for code in their development environments as per S(2)-2. As per S(4)-5 and S(4)-6, security measures are implemented in development tools to protect development environments. With these measures, code is protected from attackers through the protection of development environments.
	• **Unauthorized third-party software incorporation** Vulnerable third-party source code or binaries or software or components of unknown origin are intentionally or accidentally incorporated.	Businesses procure secure software components as per S(2)-1. The origins of components are managed in the respective software releases as per S(2)-2. With these measures, the incorporation of inappropriate third-party software is prevented.

System life cycle	Outline of the threat	Reason for requirements against the threat
Design - Testing	• **Unauthorized incorporation during build** An attacker exploits a flaw in a build process, and unauthorized software is incorporated into a product component. (Example: Inappropriate compiler option)	Businesses use secure build tools to build products as per S(1)-2. Security measures are implemented in development tools for the build environment as per S(4)-5. With these measures, unauthorized incorporation during builds is prevented.
	• **Development process dependent on individual skills with low accuracy and reproducibility** Owing to PDCA procedures not being followed and the development (implementation) process being excessively dependent on individual work, accuracy and reproducibility are reduced, resulting in potential security issues in a build environment. (Example: Local build dependent on individual skills)	Businesses use secure build tools to build products as per S(1)-2. Appropriate development toolchains are used, as per S(4)-5. With these measures, development work dependent on individual skills is avoided.
	• **Omission of review/analysis** Vulnerabilities remain unsolved owing to insufficient review and analysis of the code to identify vulnerabilities and conform to standards. (Example: Lack of vulnerability testing and scanning)	Businesses conduct reviews at the design stage as per S(1)-1. Testing is implemented as per S(1)-3. Various forms of code are reviewed as per S(1)-2 and feedback provided to process. With these measures, review and analysis are implemented to an appropriate extent.
	• **Inappropriate development process** No PDCA procedure is established and a low-quality development process is adopted. In addition, excessive prioritization of time to market and cost reduction causes new development processes and approaches to be forcibly adopted, resulting in potential security problems. (Example: Weak development standards)	Businesses prepare a development policy as per S(4)-2 and arrange development standards as per S(4)-4. With these measures, secure development processes are maintained.
	• **Unintentional information leakage** Information is leaked unintentionally. (Example: Carelessness of a developer or an inappropriate development environment)	Businesses strive for improvement of personnel skills through training, etc., as per S(4)-1. A secure software development infrastructure is arranged as per S(4)-2. A software service operation policy is established as per S(4)-3. As per S(4)-5 and S(4)-6, security measures are implemented in development tools to protect development environments. With these measures, information leakage from the human and environmental perspectives is reduced.
	• **Inappropriate service use (cloud only)** (Owing to schedule and cost constraints) SaaS services in which security is not considered are introduced or implemented. (Example: Different modules in a single SaaS solution have different security requirements, causing insufficient verification of all components)	Businesses introduce services consisting of appropriate software components as per S(2)-1. A process is arranged for dealing with risks that do not meet security requirements as per S(2)-3. With these measures, appropriate services are introduced/used.

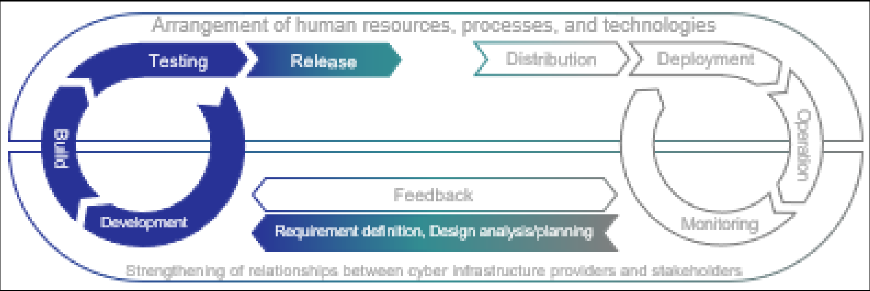
System life cycle	Outline of the threat	Reason for requirements against the threat
Distribution	• **Unauthorized incorporation during distribution** Malicious software is injected into an original software package, update program, or upgrade product distributed to customers through a software distribution route or delivery mechanism. (Example: Program tampering, document tampering)	Businesses provide information and mechanisms that allow users to start using software securely as per S(2)-4. With these measures, fraud is prevented during distribution.
Operation	• **Denial of service** An attacker stops an external service such as SaaS, or stops supply of security patches, etc.	Businesses arrange a software service operation policy as per S(4)-3. Businesses prepare a monitoring environment for service operation as per S(1)-4. Risk response to vulnerabilities is implemented as per S(3)-2. With these measures, vulnerabilities that are causes are addressed, and denial of service is detected and its impact reduced.
	• **Unauthorized archive manipulation** Archives are manipulated, overwritten, or destroyed, either unintentionally by a developer or intentionally by an attacker. It becomes difficult to analyze and respond to vulnerabilities discovered after a release.	Businesses protect archives as per S(2)-2. With these measures, unauthorized manipulations of archives are prevented.
	• **Vulnerabilities left unsolved** Use of software continues without information regarding discovered software vulnerabilities being communicated to customers.	Businesses establish a vulnerability disclosure policy as per S(3)-1. Users are provided with information necessary for measures against vulnerabilities as per S(2)-4. Businesses address vulnerabilities, including their root causes, as per S(3)-2 and S(3)-3. Businesses introduce a software component update process as per S(2)-1. Businesses provide customers with appropriate information and measures to address vulnerabilities, aiming to eliminate vulnerabilities.
	• **Incorrect configuration/settings** When software is used without appropriate configuration or settings, vulnerabilities become apparent. (Example: Full access permitted by default, execution of software whose authenticity cannot be confirmed, etc.)	Businesses promote the development of software that is secure by default as per S(1)-2. Businesses provide users with secure configurations and usage methods as per S(2)-4. With these measures, the use of appropriately configured software is promoted.
Disposal	• **Information leakage and unauthorized access through disposal** Confidential information such as source code is stolen by an attacker through discarded equipment that retains confidential information stored in it, and unauthorized access to it is gained through equipment that is left unused and to be discarded.	Businesses arrange a software service operation policy as per S(4)-2 and S(4)-3. With these measures, equipment intended to be discarded is appropriately disposed of.

System life cycle	Outline of the threat	Reason for requirements against the threat
Common to life cycles	• **Processes and resources unarranged** Secure software cycles and supply chains are not maintained because processes and resources (people, things, and money) are not arranged. (Example: Insufficient training to conduct threat or risk assessments, security is not taken up as a management issue, excessive workload, etc.)	Businesses arrange human resources as per S(4)-1. A development policy is established as per S(4)-2. A software service operation policy is established as per S(4)-3. Processes and resource systems are audited as per S(4)-4. Customers appropriately determine systems of businesses as per S(6)-1. Appropriate budgets are ensured as per S(6)-2 . With these measures, people, materials, and money necessary to ensure security are arranged, such as development infrastructures and security requirements and standards for development processes.
	• **Inadequate information and asset management** Security initiatives are not initiated because the necessary information is not grasped.	Businesses promote information sharing as per S(5)-1. Businesses promote proactive utilization of cooperative structures as per S(5)-2. With these measures, the collection and management of information necessary for security measures are promoted.
	• **Incomplete agreement chains** Security requirements are not satisfied or requests are rejected because contractual agreements regarding security between suppliers, third-party suppliers, and customers do not properly link up.	Businesses seek agreement on appropriate security requirements among parties concerned as per S(2)-3. Risk response is maintained as per S(1)-1. Customers proactively manage risks as per S(6)-1. Customers disclose in advance security practices required of businesses as per S(6)-2. With these measures, the parties concerned agree on appropriate security requirements.
	• **Inadequate selection conditions** Characteristics of suppliers are not considered when a supplier (subcontractor) and software are selected. (Example: Past performance, etc.)	Businesses agree on security requirements among parties concerned as per S(2)-3 and include them in contracts. With these measures, appropriate suppliers are selected.

5.4. Examples of measures implemented to meet requirements

Examples of measures to be implemented to meet requirements (itemized requirements) are described below as reference information. For measures to be organizationally implemented in the target software to meet requirements, organizationally appropriate methods must be selected and applied. These examples are provided as reference information so that such measures can be envisioned. Consequently, they do not comprehensively represent the measures to be implemented for the requirements (itemized requirements). When considering measures to meet these requirements, it is recommended to refer to 5.5, as necessary.

(1) Secure design, development, supply, and operation

S (1)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Risk assessment during design and tracking of countermeasures Analyze and assess the risks of software to be developed in accordance with the principles of "secure by design" and "secure by default," track risk responses, security requirements, and design decisions, and maintain countermeasures.		
☐ **S(1)-1.1 Risk-based security requirements definition** Perform risk-based analysis and assessment of the software to be developed or the system/service using the software, and define security requirements that serve as mitigation measures.		
Examples of measures	• To improve the effectiveness of the risk-based approach, conduct risk assessments using risk analysis techniques using risk modeling, like attack and threat modeling. • To improve the effectiveness of the risk-based approach, train development teams or consult risk modeling experts. • Conduct more strict risk assessment for high-risk areas such as protection of confidential data and personal information, authentication, access control, and credential management. (In the case of software for a system/service) • Establish security requirements to be agreed upon with customers as company-wide rules in advance. For customers who do not provide requirements, establish appropriate security requirements through hearings. • To obtain customers' understanding of the costs associated with security measures, propose benefits obtained by improving customer's security. Simultaneously, explain the necessity of increased costs to customers, based on detailed breakdowns. • Establish cooperative frameworks for entire system life cycles, including the division of roles between the business operator and customer, commonality and standardization within industries through the arrangement of development environments and terminologies, and communication methods.	

☐ **S(1)-1.2**	**Design review** Through a review of the software design, confirm that it meets all security requirements and adequately addresses identified risks, and apply the review results.
Examples of measures	• Review software from design perspectives (architecture, design, critical code, vulnerabilities, etc.) using an appropriate method for each perspective (peer review, lead review, walkthrough, static and dynamic scanning, vulnerability scanning, etc.). • Review software from various development perspectives—integrated development environment (IDE), build pipeline, and automated process instantiated in a toolchain, such as static and dynamic security—using an appropriate method for each perspective (peer review, lead review, walkthrough, static and dynamic scanning, vulnerability scanning, etc.).
☐ **S(1)-1.3**	**Risk response records** Maintain records of design decisions, responses to risks, and approved exceptional measures for audit and maintenance purposes throughout the software life cycle.
Examples of measures	• Record responses to respective risks, including decisions on design, how risk mitigation was achieved, and the rationale for approved exceptions to security requirements. • Maintain records of responses to respective risks.
☐ **S(1)-1.4**	**Periodic risk-based review** Review all approved exceptions to security requirements and software design, as well as the results of the risk-based analysis and assessment created during the software design, and periodically check whether risks are being addressed appropriately.
Examples of measures	• Regularly re-evaluate all approved exceptions and implement appropriate changes as necessary. • Review risk models to check periodically whether risks are addressed appropriately and implement changes as necessary (SP800-218 PW.1.2 notional implementation example).

■ Threat modeling and risk management

Threat modeling is an analytical technique for identifying potential threats and vulnerabilities in software and studying security measures to be implemented to make connections to risk management.

Intended software and assets to be protected are clarified, and threats and vulnerabilities that adversely affect assets are analyzed.

Various frameworks have been published, a typical one is the STRIDE model developed by Microsoft. This model is used as a methodology for identifying threats and studying security measures from the perspectives of "spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege." (Related requirement: S(1)-1.1)

Column	Guidance on "bad practices"

Various organizations have arranged best practices for software security, but as a contrary concept, the US Cybersecurity and Infrastructure Security Agency (CISA) and Federal Bureau of Investigation (FBI) have published "Product Security Bad Practices Guidance." The guidance describes inappropriate product security practices that are considered as high risk for software vendors and provides recommendations for software vendors to mitigate these risks. These are divided into three categories, as described below. The first and second ones are most relevant to the Guidelines.

[1] Product characteristics:

These relate to the observable security quality of software, such as development in a language that is not memory safe. In direct relation to the Guidelines, the release of a software product that contains components with known vulnerabilities and the use of vulnerable open-source software fall under this category.

[2] Organizational processes and policies:

These relate to transparency assurance for software security, such as failing to publish vulnerability disclosure policies.

[3] Security functions:

These relate to security functions that software products should have, and a lack of multi-factor authentication and logging functions, are described.

■ Entities that perform risk assessments of software to be developed

"Software to be developed" includes software that realizes the functionality of a system/service, software to be embedded in an IoT device, and firmware to be installed on a chip.

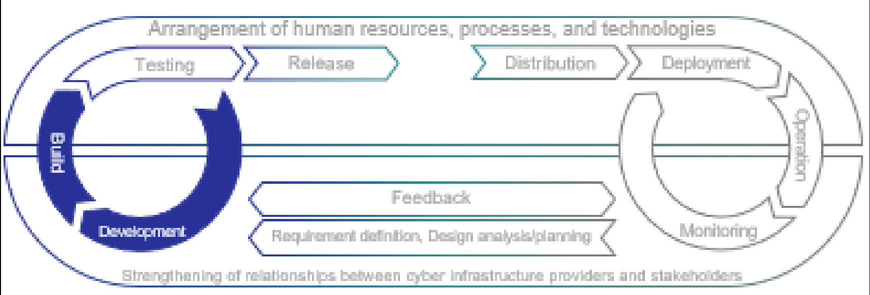
Entities that operate a system/service carry out risk modeling and analysis/assessment at the system/service level. In addition, entities that design and manufacture IoT devices and chips carry out risk modeling and analysis/assessment at the IoT device and chip levels.

In addition, risks to be focused on in S(1)-1 are risks related to software to be developed, and developers carry out risk modeling and analysis/assessment proactively. For dedicated software, higher-level risks for which a usage environment is identified must be considered. With respect to general-purpose software, risks based on the usage of software in expected usage environments must be considered.

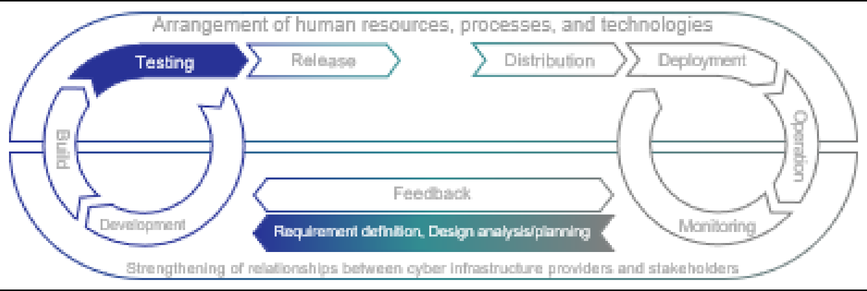
■ Approach for costs associated with security measures

To obtain customers' understanding of the costs associated with security measures, it is necessary to create proposals that are based on not only profit distribution of cyber infrastructure providers for themselves but also security improvements of customers. In addition, it is necessary to provide educational activities for customers regarding costs, details of increased costs (estimation of required costs in the case of system development or renovation, and the addition of service menus applied in the case of a cloud service), and the necessity of accountability to customers.

It is important that both customers and cyber infrastructure providers share the same understanding regarding the necessity and costs of security measures, and it is desirable to foster understanding through the division of roles between customer and cyber infrastructure provider, commonality and standardization within the industry through the establishment of development environments and terminology, and communication throughout the life cycles of intended systems. (Related requirement: Entire S(1)-1)

S (1)-2	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Secure build Define secure coding and system construction processes that are appropriate for development languages and environments, and generate and build code accordingly. Review and analyze code, including configurations, and feed back the results to the process.		
☐ **S(1)-2.1 Definition of secure development process** Define processes related to secure coding, secure build, and secure by default by considering secure coding perspectives, the build timing and method, the use of automation tools, and training.		
Examples of measures	• Check for vulnerabilities that are common to development languages and environments and prevent these vulnerabilities from being incorporated into a process. • ntroduce automation support (quality improvement) into development methods and environments. • Apply configuration management tools to manage intermediate deliverables and configuration baselines in development. • Provide appropriate training before using secure coding techniques and development environments equipped with automated features. (SP800-218 PW.5.1 notional implementation example) • Implement secure default configurations, store default configurations in a usable format, and enable changes in accordance with change management practices. • Document secure settings and guidance on operations for software users (such as system administrators). • Introduce automation support with AI support in development techniques and environments (such as quality improvement through AI application to static analysis).	

☐ S(1)-2.2	**Secure build** Generate and build code using a compiler, an interpreter, and build tools that provide functions to improve the security of executable formats.
Examples of measures	• Determine functions and configurations of compilers, interpreters, and build tools and make approved configurations available in the form of configuration as code (CaC). • Establish a change management process to deploy/update compilers, interpreters, and build tools and periodically verify their authenticity and integrity. • Apply protected configurations to virtualization technologies such as containers used to deploy software.
☐ S(1)-2.3	**Verification and feedback** Identify root causes of problems discovered through verification by review and analysis, and then feed the results back to the processes.
Examples of measures	• Select a method for reviewing and analyzing codes depending on the stage of the software life cycle. (SP800-218 PW.7.1 notional implementation example) • Perform code reviews and analyses based on in-house secure coding standards, and record and prioritize all discovered problems and recommended solutions in a development team's workflow or problem tracking system. (SP800-218 PW.7.2 task)Obtain assistance from expert reviewers to check whether backdoors or other malicious codes are present. • When a tool such as a static code analysis tool is used, document the results of the analysis. • Use static analysis tools to check codes for vulnerabilities and compliance with in-house secure coding standards automatically, and review any issues reported by a tool and solve them as necessary. (SP800-218 PW.7.2 notional implementation example) • Verify compliance with security requirements through review and analysis, identify and document the root causes of any issues found, and feed back the results of the response to processes for secure coding, secure build, and secure by default. (Derived from statement)
☐ S(1)-2.4	**Codebases** For objects subject to review and analysis, not only source codes but also codes in various formats (such as configuration files) that the organization determines to be readable should be targets.
Examples of measures	• In intended settings of development environments subject to review, include compiler configurations, development languages and environments to prevent common vulnerabilities and weaknesses, and third-party codes and reusable code modules written in-house, as needed.

S (1)-3	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Testing

Design and implement vulnerability and penetration testing as well as functional testing to identify vulnerabilities not identified in the review and analysis up to the build phase, and implement countermeasures against identified vulnerabilities.

☐ **S(1)-3.1 Test planning**

Based on threat models and risk analysis, determine a test scope and test method, and develop a test plan.

Examples of measures

- Determine whether executable code needs to be tested to identify vulnerabilities not identified in reviews, analysis, or testing. (SP800-218 PW.8.1 task)
- When testing executable code, determine the scopes and methods of the testing and develop test plans. (SP800-218 PW.8.1 task)
- In those to be tested, include binaries, directly executed bytecodes, source codes, and other forms of code and software that organizations regard as executable.
- In those to be tested for code, include third-party executable codes and in-house created reusable executable code modules as necessary.

(In the case of software for a system/service)

- When a client business distributes testing guidelines or templates, develop a test plan including these.

☐ **S(1)-3.2 Test method**

Include functional testing, vulnerability testing, fuzzing, penetration testing, etc. in the test method.

Examples of measures

- Include test methods to verify that the respective settings, including default settings, function as expected and do not inadvertently cause security vulnerabilities or operational problems.
- Include fuzz testing in the testing methodology to identify problems with input/output processing within the software.
- Include penetration testing to simulate manners in which an attacker breaches a software in high-risk scenarios.
- Integrate static and dynamic vulnerability tests, as well as regression tests to remove negative effects of modifications, into automated test suites of projects.

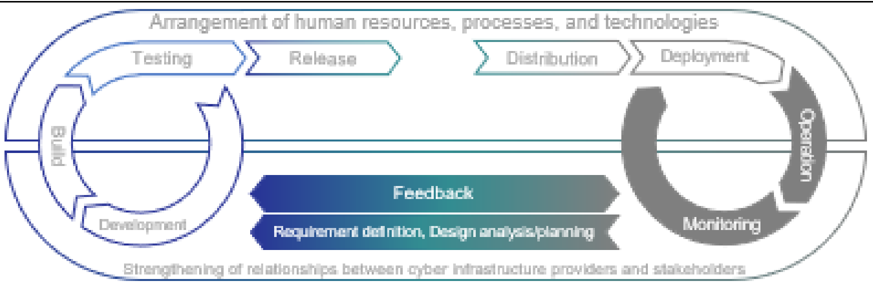
☐ **S(1)-3.3 Test implementation**

Design and implement tests according to the test plan, and document the test results.

Examples of measures

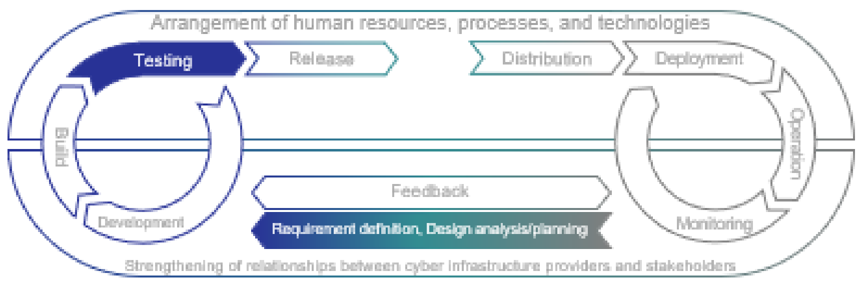
- Conduct tests in which a practical production environment is considered according to manuals to make it possible to confirm that the respective settings, including default settings, function as expected.

☐ **S(1)-3.4 Responses to problems** Incorporate all problems identified through testing and recommended countermeasures into the development team's workflows to solve them.	
Examples of measures	• Record all problems discovered through testing and recommended countermeasures in the development team's workflows or problem tracking system, and prioritize and document them. (SP800-218 PW.8.2 task) • Identify and record the root causes of discovered problems. (SP800-218 PW.8.2 notional implementation example) • Organize the results of the vulnerability analysis, vulnerability risk mitigation measures, and tool analysis so that they can be presented upon request by a reviewer.

S (1)-4	**Developer** Supplier **Operator** Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Monitoring of services Arrange a process and system to ensure that software protects and maintains information assets and is consistent with the environment in which it is implemented (network, platform, service, etc.) and implement them.		
☐ **S(1)-4.1 Asset management** Operators arrange asset management procedures and asset lists related to assets handled by systems and services as well as assets that constitute the systems and services.		
Examples of measures	• Integrate change management and configuration management into asset management for systems and services to maintain secure configurations. • Establish and perform procedures for maintaining the security of systems and services based on security policies.	
☐ **S(1)-4.2 Development of a monitoring environment** Operators separate systems appropriately to minimize the potential impact of a risk when it occurs, and arrange a monitoring environment to monitor risks that are important to protect assets by means of software.		
Examples of measures	• Use systems for managing services for a dedicated purpose only so that they do not intermingle with other tasks. • Apply diagnostic tools to determine important risks.	
☐ **S(1)-4.3 Arrangement of a security mechanism** An appropriate security mechanism is arranged that allows software and systems and services to which the software is applied to protect and monitor the confidentiality and integrity of information assets and data in operating environments or resources such as digital infrastructure.		
Examples of measures	• Use firewalls, encryption, signatures, etc. as mechanisms for protecting confidentiality and integrity.	

☐ **S(1)-4.4 Monitoring and evaluation** Operators monitor the operation of mechanisms applied to software that provides important services, periodically conduct security assessments, and integrate them into the risk management framework of the organization.	
Examples of measures	• Monitor the status of the operation of software mechanisms applied to critical services and ensure that they are protected and maintained consistently with networks, platforms, and other interlocking services related to the operation of the software. (Derived from statement) • Include the time when a new system is introduced and the time when major changes are made to an operational system in the timing of security evaluation related to systems and services.

(2) Life cycle management and assurance of transparency

S (2)-1	**Developer** Supplier Operator Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Arrangement of secure software components Verify that commercial, open-source, and other third-party software components procured from outside comply with the defined in-house requirements throughout their life cycles.		
☐ **S(2)-1.1 Arrangement of software components** With respect to commercial, open-source, and other third-party software components procured from outside, adopt those that are highly secure and meet the defined in-house requirements.		
Examples of measures	• Review and evaluate third-party software components (including software libraries, modules, middleware, frameworks, etc. that provide standardized security functions and services such as cryptographic modules and standard protocols), assuming its usage environment. • Determine secure configurations of third-party software components and make it easy for developers to use the configurations with CaC, etc. (SP800-218 PW.4.1 notional implementation example) • To verify the security of third-party software components in an assumed usage environment, create builds from source code (including security scan), static analysis (binary scan), dynamic analysis, etc. as necessary. (In the case of software for a system/service) • Submit a self-conformance certificate indicating compliance with the SSDF upon customer request, and submit an SBOM as a deliverable indicating the compliance, as necessary.	

☐ **S(2)-1.2**	**Development and maintenance of software components** When the software components are not procured from outside, develop highly secure software components in-house in accordance with established in-house security standards and practices, and maintain them.
Examples of measures	• When developing and maintaining components, conduct secure software development in accordance with security practices established in-house. (SP800-218 PW.4.2 notional implementation example) • Determine secure configurations for developed software components and make them easily usable for developers through CaC, etc. (SP800-218 PW.4.2 notional implementation example)
☐ **S(2)-1.3**	**Risk assessment of software components** Acquire and analyze information regarding locations from where the respective software components are obtained and assess the risks resulting from the components.
Examples of measures	• Maintain a list of commercial software components and component versions approved in-house along with their provenance data (e.g., SBOM). (SP800-218 PW.4.1 notional implementation example) • Perform a configuration analysis (source code, binary code) of the respective software components and maintain a repository to make secure configurations easily usable. • Acquire and analyze the provenance information of the respective software components (e.g., SBOM, source configuration analysis, binary software configuration analysis), and evaluate risks that components may pose. (SP800-218 PW.4.1 notional implementation example) • Verify and confirm the integrity of the software components by making use of digital signatures or other mechanisms. In this manner, identify and verify the certificates used, and verify the cryptographic standards used. • Share the management of source codes, configuration information, and change information on supply chains, and share SBOMs as necessary.
☐ **S(2)-1.4**	**Confirmation of publicly known vulnerabilities of software components** Regularly check for publicly known vulnerabilities and periods during which respective software components are supported.
Examples of measures	• For conducting regular checks for publicly known vulnerabilities and support periods of the respective software components, consider the utilization of external diagnostic or audit services. • Incorporate the automated detection of known vulnerabilities in deployed software components into toolchains.
☐ **S(2)-1.5**	**Updating of software components** Implement a process to update the respective software components to the new version securely.
Examples of measures	• Implement a process to update the respective deployed software components to the new version, and retain the older versions of the software components until all migrations from those versions are successfully completed. (SP800-218 PW.4.1 notional implementation example) • For any vulnerabilities (including publicly known vulnerabilities) that are found in the respective introduced software components, share information on the supply chains of the software components and solve them promptly by applying patches, etc. • If the integrity or origin of an acquired binary cannot be confirmed, verify the integrity and origin of the source code and build a binary from the source code. (SP800-218 PW.4.1 notional implementation example)

■ Purpose of procuring secure software components and the need for information sharing

Instead of developing functions individually from scratch, it is possible to reduce risks of vulnerabilities by reusing existing software that ensures security, such as existing system components that are sufficiently secured or standardized software components (log management, access control, etc. that comply with standards).

Countermeasures against software vulnerabilities should be implemented as far as possible during the development stage, and it is necessary to respond promptly to any remaining vulnerabilities discovered thereafter by applying patches, etc. during the subsequent life cycle and in the supply chain. To achieve this, it is necessary to realize information sharing regarding software through configuration and change management in a sustainably maintained software development and maintenance environment based on an SSDF and cybersecurity-supply chain risk management (C-SCRM). (Related Requirement: S(2)-1.1)

■ Towards building a secure software distribution mechanism by using and sharing SBOMs

When a vulnerability is identified in a software procured from a developer or supplier, in order for the procurer (customer) to fix the software, it is recommended to obtain the source code and SBOM of the software and perform configuration and change management on them.

In addition, when standard mechanisms for distributing vulnerability-managed software (such as binaries and IoT devices with embedded software) can be realized—such as by having suppliers/developers manage source codes and SBOMs and appropriately manage configurations and change management while retaining rights to source codes, and by making it possible to trace evidence as necessary—the distribution of source codes and the SBOM may not be required with delivery. Furthermore, to consider rapid identification of the impact of vulnerabilities, it may be more effective to establish a mechanism for tracking the traceability of products developed in-house (into which they are embedded) or using a framework such as the Binding Operational Directive (BOD) by CISA or the EU CRA (in which the government requests critical infrastructure operators), or customers/operation organizations can request cyber infrastructure providers to report whether there are vulnerabilities.

As described above, the need for utilizing and sharing SBOMs is increasing, and some industries are considering unified rules. To introduce SBOMs and share them fully between organizations, it is desirable to consider and build a system with reference to the "Guidelines on Introduction of Software Bill of Materials (SBOM) for Software Management ver. 2.0" published by the Ministry of Economy, Trade and Industry on August 29, 2024. Specifically, it is important to fully understand the feasibility and constraints of a system for secure software distribution including SBOMs and the priority of efforts, and then determine a system for secure software distribution and create agreements through contracts, etc. (Related requirement: S(2)-1.3)

■ Requirements for software introduced by government-related agencies in Europe and the US

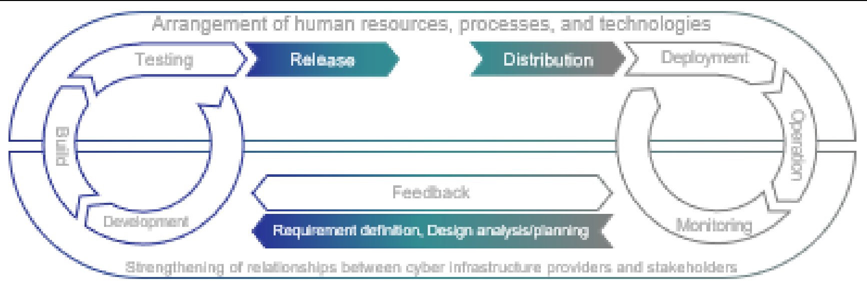
Initiatives for SSDFs are progressing mainly in Europe and the US. In the US, the Office of Management and Budget (OMB) has announced "M-22-18 Enhancing the Security of the Software Supply Chain through Secure Software Development Practices" (and M-23-16, the updated version of the same).

This document requires federal agencies to employ software vendors that can certify that they can implement the SSDF SP800-218. In addition, it requires software vendors to submit a self-certification of conformance to certify their implementation of the SSDF and an SBOM as a deliverable to demonstrate compliance, if necessary.

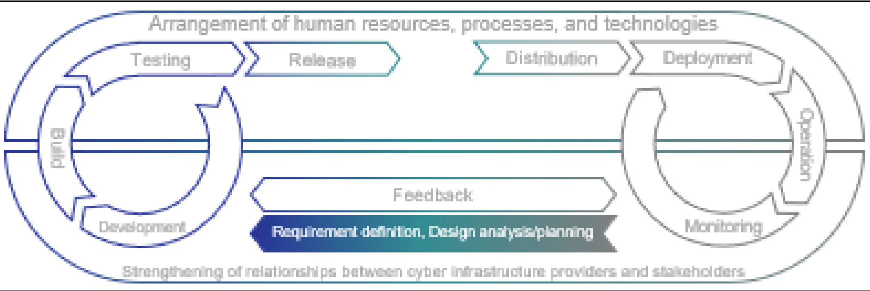
A self-certification of conformance is a document that proves compliance with the SSDF based on EO14028, and certifies processes and procedures for continuous secure software development, such as vulnerability disclosure and response. (Related requirement: S(2)-1.1)

(Reference) On page 31 of the document on the Software Task Force of the Ministry of Economy, Trade and Industry, in the following URL, there is a description of the SSDF self-certification and SBOM requirements in the OMB memorandum (M-22-18).

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/010_03_00.pdf

S (2)-2	**Developer** **Supplier** Operator Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Secure archiving of release files and data Archive necessary files and data to be retained during software release and restrict access to only necessary personnel, tools, and services. Collect, protect, maintain, and share provenance data for all components of the respective releases through the gradual adoption of SBOMs.		
☐ **S(2)-2.1 Protection of codebases** To protect codebases in all forms from unauthorized access and tampering, store the codes and configuration information in a repository and implement access control based on the principle of least privilege so that only authorized personnel, tools, and services can access it.		
Examples of measures	• Store all forms of code, including source codes, executable codes, settings, resource files, container images, and CaC, in codebase repositories. This applies to open-source and language class component groups, integrity verification information, provenance data, etc. • Use cryptographic techniques (e.g., code signing, commit signing, hashing) to protect the authenticity and integrity of source codes and executable code files. • Have a third party review all changes made to the code and code owners approve them.	

☐ **S(2)-2.2 Archiving of releases** Archive the respective software releases to protect them so that vulnerabilities identified following release can be analyzed and identified.	
Examples of measures	• Store release files, associated images, etc. in repositories according to established organizational policies. Allow read-only access for necessary personnel and prohibit access by others. (SP800-218 PS.3.1 notional implementation example) • When enhancing functionality, store associated codes and executable files, and check and approve all changes.
☐ **S(2)-2.3 Sharing of release provenance data** Collect, protect, maintain, and share provenance data for all components of the respective software releases.	
Examples of measures	• Make provenance data available to in-house operations and response teams that receive and acquire software using SBOMs, etc. • Document all third-party components acquired directly by developers and incorporated into the software, and employ measures to trace their original sources as far as possible. • Scan binaries created by third parties and conduct risk assessments for the security of components created by third parties. • Implement a check system for open-source libraries and check them regularly. • Both suppliers and developers work together to ensure the integrity of signing servers for integrity verification.

S (2)-3	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Establishment of security requirements among stakeholders Establish security requirements to be agreed upon among the parties involved and include them in contracts or policies to be shared.		

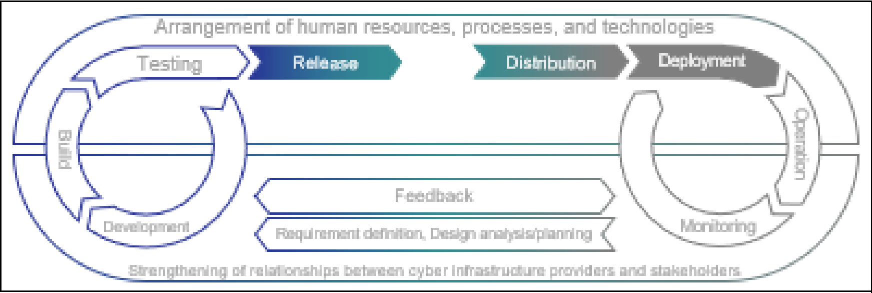
☐ **S(2)-3.1**	**Agreement on security requirements** Include explicit security requirements in contracts or policies to be shared with third parties that provide software (including commercial software components for use in in-house software) or services.
Examples of measures	• Examples of requirements to be included in contracts or policies with third parties (suppliers) are as follows: ➢ Monitoring and disclosure of supplier's information security compliance (software security requirements) ➢ Regulations for sharing information on potential problems with suppliers ➢ Implementation of secure development processes (including process verification, penetration testing, etc. by third parties) ➢ Vulnerability management (including vulnerability disclosure and patch management) ➢ Provision of SBOMs ➢ Implementation of processes to ensure the authenticity of components from suppliers (protection against unauthorized access during transmission of data related to supply chains) ➢ Response to vulnerabilities related to products and services of suppliers ➢ Assurance of availability of suppliers and measures for recovery ➢ Provision of support, definition of SLA, complaint handling ➢ Others, such as definitions of responsibilities and roles of both parties, and requirements for contract completion and termination
☐ **S(2)-3.2**	**Responses to supply chain security requirements** Respond to supply chain security requirements equivalent to those adopted by the organization that receives or acquires third-party software or services that it provides.
Examples of measures	• Maintain a process for selecting suppliers of components made by a third party based on supply chain security requirements and obtain evidence of selection. (In the case of software for a system/service) • To demonstrate that specifications for software quality that customers require are satisfied, adopt a mechanism to run customer-specified source code diagnostic tools and submit evidence.
☐ **S(2)-3.3**	**Establishment of a response process for risks that do not meet security requirements** Arrange a process to respond to risks in the case in which there are security requirements that third-party software or services to be received or acquired do not meet.
Examples of measures	• Determine acquisition strategies and procedures to reduce supply chain risks (do not clarify purchase purposes, select reliable distribution destinations, provide incentives to suppliers with good contract terms and management, etc.). • Verify and update SBOMs obtained from third parties. (In the case of software for a system/service) • Implement and provide configuration management, change management, hardening of development and maintenance environments, etc. to manage and supervise supply chains.

■ Support for / cooperation with suppliers and developers who are contractors

To establish and maintain security through the entire supply chain, customers who are clients must provide as much support and management as possible to meet the agreed upon supply chain security requirements as part of support for suppliers and developers and cooperate with them. For example, the following support and management can be considered:

- Provide suppliers and developers with a development environment or allow them to use it. Permit the contractor's personnel to use the development environment after they take a course and pass a test.
- Make agreements in stages, starting with important security requirements.
- Prime providers support and manage contractors. In cases in which multiple cyber infrastructure providers are involved, customers or a consulting company that oversees and handles entire upstream processes participates to ensure that all vulnerabilities are identified and not included.
- When the sub-contractor is a subsidiary, the parent company may control the supply chain security, but as it is necessary to be careful about the sharing of benefits, financial and IT departments participate to achieve an overall balance.
- Set a grace period for contractors who cannot immediately provide a response when management standards are revised.

Thus, in responses to supply chain security requirements, effort levels vary between supply chain layers, such as prime contractors, sub-contractors, and distributors. Therefore, it is necessary to raise the levels of basic efforts of each stakeholder in supply chain considering the current situation they are in. In this case, it is important to decide on the division of roles and items to be implemented at the time of contract to deepen the cooperative relationship between stakeholders. In the future, it is desirable to arrange guidelines through deeper discussions with industry groups and public institutions, and to organize how to determine the scopes of responsibilities and perspectives through future efforts by the involved parties. (Related requirement: Entire S(2)-3)

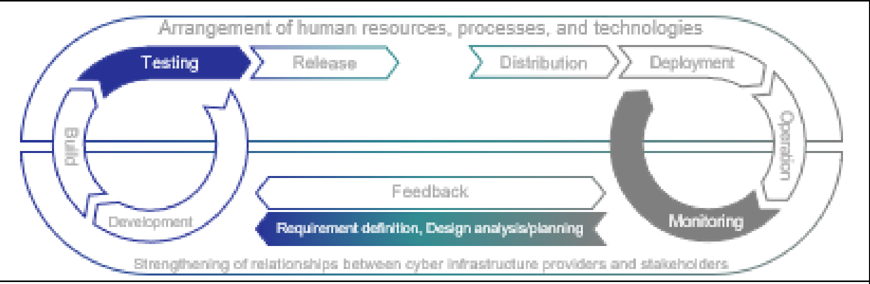
S (2)-4	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Appropriate information provision to users Ensure that software users can use guidance that facilitates secure use throughout the entire software life cycle—from introduction and installation to operation and termination of use.		

☐ **S(2)-4.1**	**Secure introduction, configuration, operation, modification, disposal, and termination** Ensure that software users can continuously use information for securely introducing, configuring, and operating software, as well as information related to the impact of changes, disposal, termination of provision, and termination of use.
Examples of measures	• Implement secure default settings (or groups of default settings, if applicable). (SP800-218 PW.9.2 task) • Include the following in guidance for software users (system administrators, etc.): ➢ Secure introduction procedures for initial installation, installation of additional components, updates, and patches, and procedures for secure configurations ➢ Software integrity verification information and configuration guides ➢ Details of secure configuration information (purposes of the respective settings, default settings, relevance on security, potential operational impact, relationship with other settings, etc.) • Clearly communicate the decision to end software support to users (customers) and specify the scheduled date for the end of support. • Even after software is supplied, if a vulnerability is identified in a program developed in-house, provide the necessary information to ensure the safety of software users, for example, by quickly and widely providing information to relevant parties in supply chains. (In the case of software for a system/service) • Communicate realistic expectations regarding contents and duration of product support in parallel with initial system provision. • Cloud service providers offering services provide information in an easy-to-understand manner to users and provide specific information guides to operators and system builders/installers.
☐ **S(2)-4.2**	**Provision of integrity verification information** Ensure that software users can continuously use information that is necessary for verifying the integrity and completeness of the software.
Examples of measures	• Provide the following for software users (system administrators, etc.): ➢ SBOMs or information equivalent to SBOMs for software to be supplied ➢ Information for verifying that measures against tampering are appropriate in distribution channels from suppliers to customers (cryptographic hash of release files, code signatures, etc.) • Provide the following for software users (system administrators, etc.): ➢ Protection measures for distribution systems (use of trusted certificate authorities for code signing, regular review of code signing processes, other measures to protect signing environments, etc.)

■ Requirements that specify secure guidance

The Common Criteria (ISO/IEC 15408, ISO/IEC 18045), which are international standards for security evaluation of IT products, require that appropriate instructions for users regarding secure installation and use be included in manuals so that customers, who are users of IT products, can begin secure installation and operation. In addition, in CISA's "Defense against software supply chain attacks," it is indicated that a mechanism for verifying the integrity of software releases (such as protection of code signing certificates) is provided so that customers can confirm that software that they have obtained has not been tampered with. (Related requirements: S(2)-4.1)

(3) Prompt responses to remaining vulnerabilities

S (3)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Continuous vulnerability investigation Establish a policy for disclosure and remediation of software vulnerabilities, define roles, responsibilities, and processes required for the policy, and implement them.		
☐ **S(3)-1.1 Establishment of a vulnerability response system** Establish a policy for the disclosure and remediation of vulnerabilities of software products, establish a system for responses to vulnerabilities (including responses to incidents) to support the policy, and define necessary roles, responsibilities, and processes.		
Examples of measures	• Establish a product security incident response team (PSIRT) for software and arrange an incident response process related to product security. (SP800-218 RV.1.3 implementation example) • Establish clear methods and procedures for common threats that violate software product security, incident response triggers, steps, recovery time objectives, and contingency plans related to services. • Conduct periodic exercises of incident response processes.	
☐ **S(3)-1.2 Communication plan** Establish a communication plan for all stakeholders.		
Examples of measures	• Arrange a vulnerability disclosure process, including a communication plan for all stakeholders. • Arrange mechanisms (e.g., mailing lists, portals) to support easy access to disclosed vulnerability information and the import of it.	
☐ **S(3)-1.3 Vulnerability information collection** Collect new information regarding vulnerabilities through searches of public information, notifications from software users, the acquisition of external threat information, reviews of system configuration data, and other methods.		

Examples of measures	• Arrange a vulnerability information collection process. (Derived from statement) • Collect information on vulnerabilities in software and third-party components incorporated into software from public sources (monitor CVEs and third-party support channels). • Collect and investigate information regarding suspected software vulnerabilities identified by vendors of software and third-party components incorporated into software, acquirers/users of the software (e.g., customers), and third-party researchers. • Identify third-party components incorporated into software and periodically check for fixes and end-of-support dates. • By making use of threat intelligence sources, gain a better understanding of how common vulnerabilities are exploited. • Automatically review the origins and software configuration data of all software components to identify new vulnerabilities contained within them. (SP800-218 RV.1.1 notional implementation example) (In the case of software for a system/service) • Consider using tools such as SSVC to manage vulnerabilities based on response priorities (make effective use as an integrated initiative with in-house asset management).
☐ **S(3)-1.4**	**Identification of undetected vulnerabilities** Conduct software code review, analysis, and testing on an ongoing or regular basis to identify undetected vulnerabilities (including improper settings) to be solved.
Examples of measures	• Maintain a system to record and track all reports of potential software vulnerabilities. • Apply the practice specified in S(1)-2.3 (verification and feedback). • Configure toolchains to perform automated code analysis and testing periodically or continuously for all supported releases. (SP800-218 RV.1.2 notional implementation example)

■ Coordination regarding responses to and reminders for vulnerabilities

In responding to vulnerabilities, reminders for vulnerabilities along with the creation and application of patches, responses based on the position as a business (SIers, operator, sales agent, etc.) and mutual collaboration are necessary. Examples are as follows. (Related requirement: S(3)-1.2)

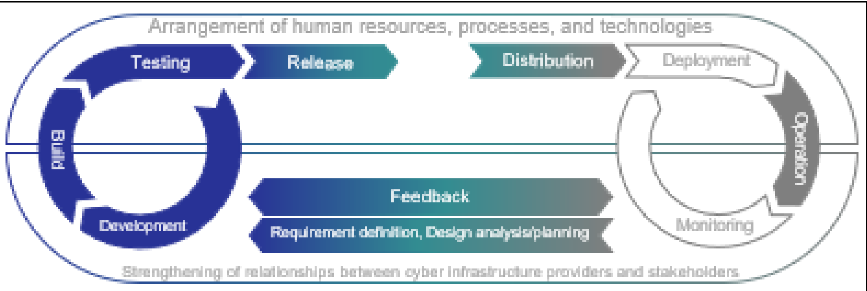
- Software manufacturers and IoT manufacturers respond to serious product vulnerabilities (create fixes) and send reminders to customers, SIers, and sales agents.
- When a software or IoT manufacturer provides information to a customer, SIers respond to serious product vulnerabilities (provide a patch to the customer and apply it upon request from the customer).
- When a software/IoT manufacturer or SIer provides information to an operation and maintenance vendor, the vendor responds to serious product vulnerabilities; they provide the customer a patch and apply it upon request from the customer.
- When a software or IoT manufacturer provides information for a distributor, the distributor notifies the customer of serious product vulnerabilities.

■ When responses to system/service software vulnerabilities are inadequate

In some legacy systems that are operated in closed environments, the need to collect vulnerability information on software that is a system component or to take measures against vulnerabilities is not recognized. Moreover, in some cases, management of assets, such as software components, which is necessary as a prerequisite for vulnerability management of software that constitutes a system, is inadequate.

As described above, in systems and services, when responses to software vulnerabilities are not adequate, measures to address vulnerabilities in conjunction with existing incident responses and contingency plans can be considered. (Related requirements: S(3)-1 in general)

- Arrange a system for coordination with suppliers regarding the provision, implementation, and testing of incident response plans.
- Arrange contingency plans (including measures to ensure the safety of utilities such as electricity) and continuity strategies to ensure the continuity of services required to establish a system.
- Integrate the collection of vulnerability information and responses to vulnerabilities into these systems.

S (3)-2	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Responses to detected vulnerabilities Regularly create a plan to respond to risks of vulnerabilities remaining in released software and implement it.		
☐ **S(3)-2.1 Vulnerability analysis** Developers collect information necessary to understand the risks associated with the impact of each remaining vulnerability and analyze each vulnerability to plan remediations or other responses to risks.		
Examples of measures	• Quantitatively analyze risks for the respective remaining vulnerabilities based on estimates of the likelihood of exploitation, the impact if exploited, and other relevant characteristics. • Use available issue tracking software to record the respective vulnerabilities. (SP800-218 RV.2.1 notional implementation example) (In the case of software for a system/service) • To ensure the effectiveness of corrections associated with risk responses, store common specifications, design documents, and intermediate products so that modifications can be made by personnel other than the code generators; agree on specific defect warranty periods; and create a contract for modification costs individually.	

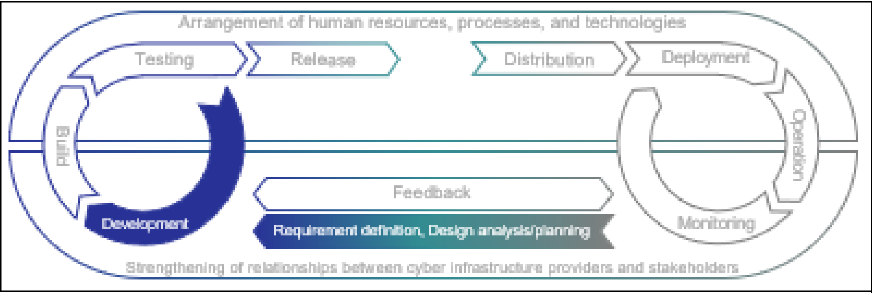
☐ S(3)-2.2	**Risk responses to vulnerabilities** Developers create a plan for risk responses for each vulnerability and implement it.
Examples of measures	• Determine risk responses on a risk-based assessment, such as whether to take measures to avoid vulnerabilities or to implement measures to repair software (including methods to temporarily mitigate vulnerabilities until a permanent solution is provided), prioritize responses to be implemented, and create plans. (SP800-218 RV.2.2 notional implementation example) • In the case of responses with implementation, document the tests and verification results. • When information regarding vulnerabilities is provided by organizations such as public institutions, create appropriate and proactive responses, including the development of required patches.
☐ S(3)-2.3	**Security recommendations** Developers prepare security recommendations, provide the information to the supplier of the released software, and create a report as specified by the relevant systems. In addition, operators implement deployment in accordance with security recommendations.
Examples of measures	• Identify vulnerabilities and components contained in software, and create documents with software configuration information. (CRA Annex I 2) • Create the necessary software correction security updates to address vulnerabilities. (CRA Annex I 2) • After a security update becomes available, prepare and disclose a security advisory for identified vulnerabilities, including descriptions of vulnerabilities, information that allows users to identify affected software, impacts of vulnerabilities, severities of vulnerabilities, and information that helps users to fix vulnerabilities. (CRA Annex I 2) • Securely distribute security updates, including patches and countermeasure procedures indicated in the security advisory, to ensure integrity and reliability. (S(3)-2.3, CRA Annex I 2) • Inform users of corrective measures that they can take to mitigate the impact of an incident without undue delay. • For vulnerabilities reported under a vulnerability reporting system, respond based on the Information Security Early Warning Partnership Guidelines. • Enable security analysts to analyze programs and report possible vulnerabilities. (SP800-218 RV.1.3 notional implementation example) • Establish a secure distribution mechanism that supports easy access and importing of disclosed vulnerability information and security updates. • To correct vulnerabilities quickly, establish an approach for appropriately and efficiently evaluating the risks of vulnerabilities, prioritizing responses, and providing a configurable automatic software update mechanism based on update strategies of customers. • Prepare a security response playbook to handle reported common vulnerabilities, zero-day vulnerability reports, vulnerabilities that are actually being exploited, and critical ongoing incidents involving multiple parties and open-source software components. (SP800-218 RV.1.3 notional implementation example)

■ Issues and responses to vulnerabilities in software for systems and services

Various issues must be considered regarding responses to vulnerabilities in software that constitutes a system or service, and thus, it is necessary to take measures according to the situation.

- For a system that provides services, there are times at which services cannot be stopped; consequently, it may be difficult to immediately perform fixes such as patch application. As a countermeasure, it may be possible to perform patch application (if permitted) with regular version upgrades, and until then, protect against vulnerabilities by employing peripheral measures for services.
- When maintaining old software packages, patch application is difficult from a compatibility perspective; therefore, a support period should be set in advance or other measures taken, and migration to software packages in which countermeasures can be taken should be encouraged.
- When adopting software with a short EOL, it may be difficult to keep up with technology and perform upgrades in some cases. In such cases, it may be beneficial to promote the development of an environment that enables efficient system upgrades with minimal time and effort, by using technologies such as containers in the medium to long term.

In modification of software as part of security measures, important issue is ensuring necessary resources, including requesting contractors to perform modifications. For example, not being able to manage costs and not being able to prepare a verification environment before applying a patch to the production environment can even increase risk. Considering such a risk, preparations should be made for necessary contents and costs, and an appropriate verification environment should be arranged. When applying for budgeting for security measures, it is absolutely necessary to explain not only the costs but also risks and their impacts to management, and it is desirable to show reasons for the necessity and appropriateness of the costs based on the risks and impacts. In this case, it is essential to reach an agreement in advance with the relevant parties on the approach to cost allocation. The timing and scope of disclosing vulnerability information may require careful consideration. (Related requirement: Entire S(3)-2)

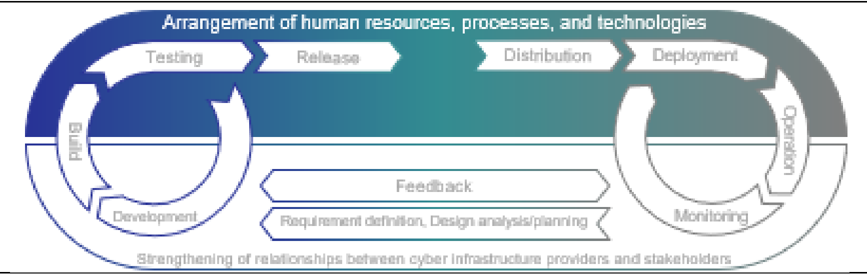
S (3)-3	Developer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Development Feedback Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Utilization of results of countermeasures to in-house process improvement Based on the analyses of vulnerabilities (root causes), review the development and operation processes so that the root causes of problems identified in software do not recur or that the possibility of their recurrence is lowered.		

☐ **S(3)-3.1 Identification of root causes**	Analyze an identified vulnerability to determine its root causes and proactively take countermeasures.
Examples of measures	• Analyze discriminated vulnerabilities that have already been identified, and analyze and record the root causes of identified problems. (SP800-218 RV.3.1 task/notional implementation example) • Analyze the root causes over time, for example, by adding mechanisms that automatically detect occurrences of causal events to toolchains to identify patterns, such as certain secure coding rules that are not observed consistently. • By using automated tools, continuously observe insecure software practices verified when a readable code is checked into a repository. • To eradicate a particular family of vulnerabilities, review software for similar vulnerabilities and make corrections in advance without waiting for external reports.
☐ **S(3)-3.2 Process improvement**	Review development and operation processes for the entire software life cycle and revise them as necessary to prevent root causes from recurring or reduce the possibility of their recurrence through software updates or new software creation.
Examples of measures	• Investigate the internal impact and implement mitigation measures for vulnerabilities to prevent or reduce the recurrence of root causes. • Review and, as necessary, update development and operation processes throughout software life cycles based on lessons learned through root cause analysis. (SP800-218 RV.3.4 task/notional implementation example) • Make use of identified root causes and corrective actions in training that helps to improve developer capabilities.

■ Development and operation should work together for vulnerability response

In preparation against the impact on a developed source code in a case where the source code or middleware is updated, sufficient operation verification of the intended software is required. In businesses in which different organizations conduct development and operation, the division of roles in operation verification becomes an issue. In particular, when transferring to an operation phase, arrangements must be made so that the operation system is composed of personnel with sufficient skills and know-how. In addition, it is necessary to arrange standards (templates and frameworks) for the operation organization to take over and manage the configuration and management information created by the development organization. (Related requirement: S(3)-3.2)

(4) Arrangement of human resources, processes, and technologies

S (4)-1	Developer Supplier Operator Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Human resources: Commitment from management and arrangement of personnel Define roles and responsibilities covering the entire software life cycle. Make management's commitment to secure development known, secure personnel for security measures, provide training to all personnel related to secure development and operation according to their levels of proficiency and role, and review it regularly.		
☐ **S(4)-1.1 Definition of roles and responsibilities** Define roles and responsibilities covering the entire software development life cycle.		
Examples of measures	• Integrate the role of security into software development teams. (Derived from statement) • Arrange roles and responsibilities to integrate supply chain management and risk management into software development processes. • Define roles and responsibilities for all involved in SDLCs, including cybersecurity staff, security champions, project managers and leaders, senior management, software developers, software testers, software assurance leaders and staff, product owners, operations and platform engineers, and procurement and inventory leaders and staff.	
☐ **S(4)-1.2 Management's commitment** Make management's commitment to secure development known to all personnel, and educate them on the importance of secure development and operation to the organization.		
Examples of measures	• Conduct trainings to ensure that all people with roles and responsibilities related to development and operation are aware of and understand the management's (top management, senior management, approval authorities, etc.) commitment to secure development and operation. (SP800-218 PO.2.3 task) • Appoint a leader or leadership team to be responsible for the entire secure software development process and implement education to raise awareness of risks and risk mitigation. (SP800-218 PO.2.3 notional implementation example) • Educate all personnel with roles and responsibilities related to development and operation about management's efforts to achieve secure development and operation as well as the importance of secure development and operation as an organization. (SP800-218 PO.2.3 notional implementation example)	
☐ **S(4)-1.3 Agreement on roles and responsibilities** Confirm that all personnel are aware of and agree to their roles and responsibilities.		
Examples of measures	• Educate individuals who have been assigned roles and those affected by upcoming changes to roles and responsibilities, and make sure that individuals understand and agree to follow their roles and responsibilities. (SP800-218 PO.2.1 notional implementation example)	

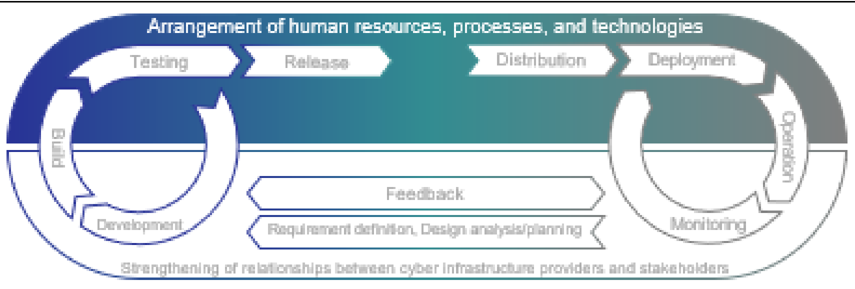
☐ **S(4)-1.4 Training for each role** Create a training plan for each role and implement it so that all personnel can be trained according to their level of proficiency and role.	
Examples of measures	• Provide role-based training for all personnel responsible for contributing to secure development. (SP800-218 PO.2.2 task) • Continuously verify the careers of personnel involved in security measures, including candidates for the roles. • Assign code jurisdiction and plan training for software developers to understand and share secure software development methods (including standardized development methods, how to use development tools in which automation is proactively used, and programming methods leveraging AI), secure coding standards, role-specific best practices, and AI-supported automation (quality improvement) methods. • In training plans, include goals by proficiency and role and a process for measuring the results.
☐ **S(4)-1.5 Review of roles and training** Review roles and training regularly.	
Examples of measures	• Review defined roles and responsibilities on a regular basis (annual, etc.) and update as necessary. (SP800-218 PO.2.1 task) • Regularly review personnel proficiency and role-based training and the assessment results, and update training as necessary. (SP800-218 PO.2.2 task/notional implementation example) • Before implementing and using new development methods and toolchains such as CI/CD pipelines based on the DevSecOps development paradigm, review training on security assurance measures for software supply chains and how to use the tools.

■ Importance of training developers on secure development

By making use of automation, human labor can be reduced and the accuracy and reproducibility of efforts throughout the entire life cycle can be improved. To obtain the benefits of automation, it is necessary to assume that actions of personnel are automated and the training for respective roles is accordingly adjusted so that the effects of automation are maximized.

As developers are directly involved in software security measures, when training for respective roles, training developers on secure development is particularly important. In the case of waterfall development, overlook of security measures in an upstream design process causes the risk of incurring high costs owing to the rework required in downstream processes. Meanwhile, in agile development, difficulty to assign dedicated or well-trained personnel results in difficulty to assign development checks to the required phases. Compared with waterfall development, there are concerns regarding risks such as the use of unintended libraries owing to the insufficient abilities or judgment of individual engineers.

(Related requirement: S(4)-1.4)

S (4)-2	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Process: Establishment of development policy and compliance with laws and regulations Comply with laws and regulations, document and maintain a security policy for in-house development infrastructures and processes, and secure necessary budgets for security establishment.		
☐ **S(4)-2.1 Definition of a software development policy** Identify all security requirements for software development infrastructures and processes (including requirements related to EOL), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.		
Examples of measures	• Define policies to protect and maintain the security of software development infrastructures and their components (including development endpoints) throughout SDLCs. (SP800-218 PO.1.1 notional implementation example) • Define policies to protect and maintain the security of software development processes and their components (including other third-party software components) throughout the SDLC. (SP800-218 PO.1.1 notional implementation example) • Make plans to maintain and recover in-house development infrastructures and processes (measures for information security control, supporting systems, processes to maintain existing measures for information security control, and control measures to compensate measures for information security control that cannot be maintained), and test, review, and evaluate the implementation. • Establish policies to verify and enforce the compliance of in-house policies considering domestic and local legal requirements at business locations (laws, administrative interpretations, etc.), industry best practices, and standards. (In the case of software for a system/service) • Define policies for acquiring systems and services (purposes, scopes, roles, responsibilities, coordination between organizations, response to compliance, etc.).	

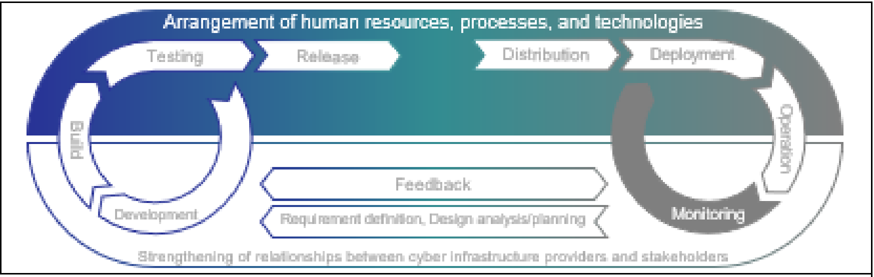
☐ S(4)-2.2	**Definition and maintenance of a software security policy** Define a policy that specifies all security requirements that must be met by the software developed by an organization, and maintain the requirements throughout the SDLC.
Examples of measures	• Include architecture and design requirements to mitigate risks, verification flow requirements at appropriate gates (checkpoints) in the life cycle, and risk response requirements for technology stacks (including language, environments, deployment models, etc.) in security requirements that software must satisfy. • Establish a policy for what should be archived during a software release (code, software package files, third-party libraries, configurations, documentation, data inventories, and other related artifacts) and how long they should be stored, based on SDLC models, software EOL, and other factors. • Review the policy periodically, when additional requirements are specified, or when incidents occur (including the discovery of vulnerabilities in-house or in released software), and communicate it to relevant parties. • Establish a process for handling exception requests to requirements (including periodic review of approved exceptions) and a process for identifying and addressing weaknesses in supply chains. • In the requirements, include an instruction to create architectures to which patches are applicable in the future. (In the case of software for a system/service) • When considering security requirements, include internal (organizational policies, business objectives, risk management strategies, etc.) and external (applicable laws and regulations, etc.) requirements.
☐ S(4)-2.3	**Sharing of cost recognition and budgeting** Secure necessary budgets to ensure security based on a policy.
Examples of measures	• Consider measures to reduce remaining cybersecurity risks below an acceptable level, secure resources (budget, personnel, etc.) required for the implementation, and then work on specific measures. (Cybersecurity Management Guidelines v3, Direction 3) (In the case of software for a system/service) • As a prerequisite for promoting the sharing of cost recognition among stakeholders, understand that leaving vulnerabilities unsolved will lead to future liabilities (damage from cyberattacks, etc.), which is a common management risk.

■ Important points to be arranged as policies

Important points to be arranged as policies are described as follows:

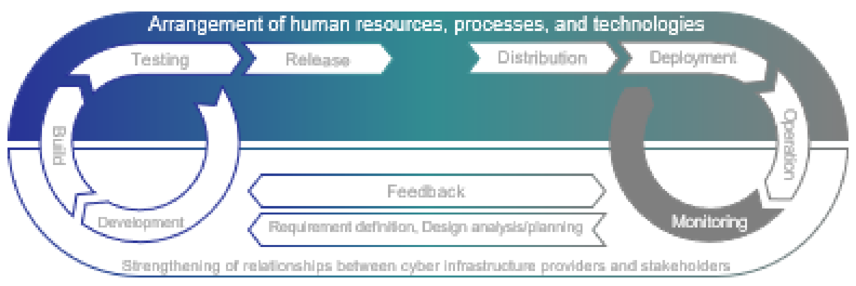
- Determine a process for handling information provided by external organizations in-house (e.g., arrange a contact point for receiving information, determine degrees of connection of received information to in-house assets and urgency, and handle information such as the formulation of a response policy).
- Manage information assets used in-house (prioritize received information based on risk management on an in-house situation basis).
- Arrange industry-specific statistical information on security, such as those regarding investments. (It is expected that management can understand the position of in-house efforts by comparing them with those of other businesses and further promote the efforts.)
- Define evaluation indicators on the effectiveness of security investments. (It is expected that effects can be quantitatively shared between the customer and business, such as by making use of reporting functions of tools invested in to visualize investment effects. However, at the moment, an easy-to-understand "visualization of investments and costs" usually turns into a problem to be studied).

When numerical targets to be achieved are organized, more realistic and precise cost calculations becomes possible, and thus, it is thought that efforts to improve the validity of cost estimates based on conformity to public numerical targets for security requirements and security baselines, such as CISA's BODs and the "minimum viable product," a security checklist jointly formulated by IT vendors, will become important in the future. (Related requirement: Entire S(4)-2)

S (4)-3	Developer Supplier **Operator** Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Process: Establishment of an operation policy and compliance with laws and regulations Comply with laws and regulations, and document and maintain all security policies on service operation infrastructures and processes to which the software is applied.		

☐ **S(4)-3.1**	**Definition of a software service operation policy** Identify all security requirements for service operation infrastructures and processes to which the software is applied (including requirements related to EOS and disposal), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.
Examples of measures	• Define policies to protect and maintain the security of software-applied service operation infrastructures, processes, and their components (including other third-party software components) throughout the SDLC. (Derived from sp800-218 PO.1.1 notional implementation example) • Create plans to maintain and recover in-house service operation infrastructures and processes (measures for information security control, supporting systems, processes to maintain existing measures for information security control, and control measures to compensate measures for information security control that cannot be maintained), and test, review, and evaluate the implementation. • Establish policies to verify and enforce compliance of in-house policies to domestic and local legal requirements at business locations (laws, administrative interpretations, etc.), industry best practices, and standards. • Establish a policy for implementing protection measures based on risk analysis according to the business size and industry. • Include a process that allows customers to link with other digital services and, if necessary, migrate to other providers that offer similar services.
☐ **S(4)-3.2**	**Definition and maintenance of a service security policy** Define a policy that specifies all security requirements that services to which the software is applied must meet, and maintain the requirements throughout the SDLC.
Examples of measures	• When considering security requirements, include requirements from within (organizational policies, business objectives, risk management strategies, etc.) and from outside (applicable laws and regulations, etc.). • Review the policy periodically or when additional requirements are specified or incidents occur (including the discovery of vulnerabilities in-house or in released software services), and communicate it to relevant parties. • Establish and follow a process for handling exception requests for requirements (including periodic review of all approved exceptions). (In the case of software for a system/service) • As a prerequisite for promoting the sharing of cost recognition among stakeholders, all stakeholders must understand that leaving vulnerabilities unsolved will lead to future liabilities (damage from cyberattacks, etc.) as a common management risk.
☐ **S(4)-3.3**	**Audit based on an operation policy** Confirm through an audit that the protection of service operation infrastructures and processes and security requirements for service are maintained throughout the SDLC in accordance with policy-based governance.
Examples of measures	• Establish a system and secure budgets so that audits do not become a mere formality. Establish governance by responding to findings received through audits of the most important and essential audit items. • Establish a mechanism for verifying the skills and capabilities of auditors from the perspective of governance.

*See reference information, "Important points to be arranged as policies," in S(4)-2.

S (4)-4	Developer Supplier Operator Customer	

Arrangement of human resources, processes, and technologies
Testing
Release
Distribution
Deployment
Build
Operation
Feedback
Requirement definition, Design analysis/planning
Development
Monitoring
Strengthening of relationships between cyber infrastructure providers and stakeholders

Process: Establishment of development and operational standards
Define security verification criteria related to software development, collect the necessary information to support the criteria, and implement processes and mechanisms for conformance. Track the status of conformance throughout the entire life cycle.

☐ **S(4)-4.1 Definition and tracking of security verification criteria**
Define software security verification criteria and track the entire SDLC.

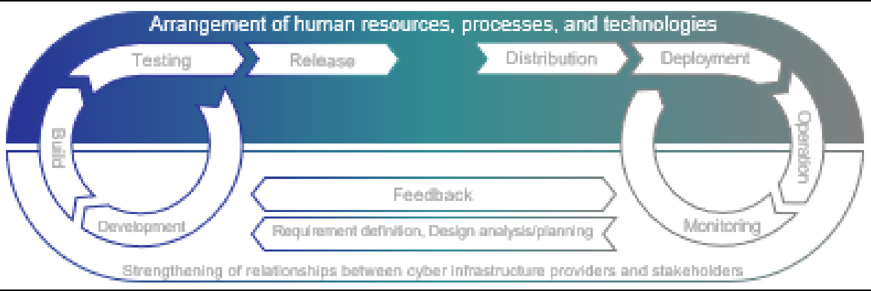
Examples of measures	• Define software security evaluation indicators based on security engineering (e.g., key performance indicators (KPIs), key risk indicators (KRIs), vulnerability severity scores, etc.) and introduce them into the development processes. • Incorporate threats, vulnerability information, and lessons learned from past projects into the security verification criteria. • Define quality indicators (e.g., no compiler errors) together to keep evidence that quality standards are met. • Record security inspection approvals, rejections, and exception requests as part of workflows and tracking systems. (SP800-218 PO.4.1 notional implementation example) • Incorporate security verification criteria into completion judgment of development workflows, check compliance status of deliverables, and use confirmed results to improve the entire development process. (In the case of software for a system/service) • Introduce KPIs that allow for constant evaluation of effectiveness in systems that support important services.

☐ **S(4)-4.2 Support for decision-making based on security verification criteria**
Implement processes and mechanisms for collecting and protecting information necessary to support decision-making based on security verification criteria.

Examples of measures	• Arrange a process to collect the necessary data to confirm standard clearing by making use of a toolchain and use it for security decision-making. (S(4)-4.2) • Deploy additional tools as necessary to support the generation and collection of information to support the criteria. (SP800-218 PO.4.2 notional implementation example) • Allow only authorized personnel to have access to the collected information and prevent it from being modified or deleted. (SP800-218 PO.4.2 notional implementation example) • Automate decision-making processes and periodically review these processes.

☐ **S(4)-4.3 Audit based on security verification criteria**
Track the entire SDLC and verify through audits that the intended effects are achieved with governance to ensure conformance to security verification criteria.

Examples of measures	• Establish a system and secure budgets so that audits do not become a mere formality. Establish governance by responding to findings received through audits of the most important and essential audit items. • Include an instruction to establish a mechanism for verifying the skills and capabilities of auditors from the perspective of governance.

S (4)-5	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	

Technology: Arrangement of secure development tools
Analyze risks throughout the SDLC and implement security measures in development tools.

☐ **S(4)-5.1 Designation of tools and toolchains**
Identify tools that are effective in mitigating specific risks and determine means of integrating toolchain components mutually.

Examples of measures

- Define toolchain categories and specify mandatory tools or types of tools to be used for the respective categories. (SP800-218 PO.3.1 notional implementation example)
- Integrate security tools into processes and toolchains.
- Define the information passed between tools and data formats used, and integrate them with toolchains or existing software development processes and workflows.
- Adopt automation techniques for managing and orchestrating tools as needed, such as to achieve build reproducibility.

☐ **S(4)-5.2 Deployment, operation, and maintenance of tools and toolchains**
Deploy, operate, and maintain tools and toolchains in accordance with security practices.

Examples of measures

- Regularly review whether tools and toolchains meet the requirements defined in-house.
- Evaluate the effects of tools on achieving security and determine their effectiveness. As expected effects, according to the purpose of application, determine the feasibility of toolchains using codebase configurations, build reproducibility, upgrade support such as vulnerability responses, whether the necessary information for verifying integrity such as origin information is available, support for toolchain automation, and responses to threats of past projects, vulnerability information, and responses to lessons learned, etc.
- Continuously research and verify the origin, integrity, vulnerabilities, and new functions of tools, and update the tools as necessary.
- When evaluating tools, conduct threat modeling and vulnerability analysis.
- Use compatibility libraries with secure third-party software toolchains as a tool security measure.

☐ **S(4)-5.3 Tool configuration and evidence generation**
Configure tools to generate evidence regarding support for secure software development practices defined in-house.

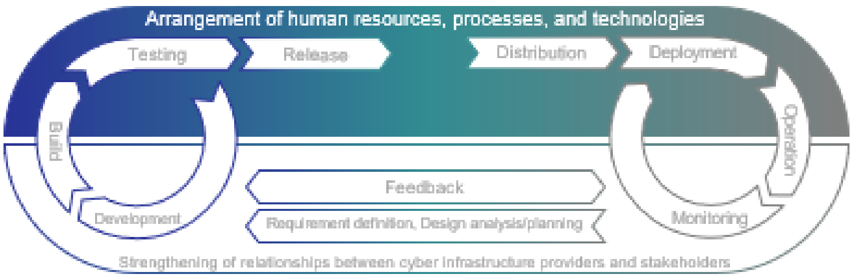
Examples of measures

- Continuously generate and monitor logs when tools are used to discover potential operational and security issues, including policy violations and abnormal behaviors.
- Use existing tools (e.g., workflow tracking, issue tracking, value stream mapping) to create an audit trail of secure development-related activities performed for the purpose of continuous improvement.
- Determine the frequencies of auditing information collected and implement the necessary processes. (SP800-218 PO.3.3 notional implementation example)

■ Supplementation of security practices for development tools

Supplementary points regarding security practices for development tools are as follows. (Related requirements: S(4)-5.2)

- When using a third-party software component such as OSS as a development environment (including a development tool), collect vulnerability information and confirm its origin.
- In the case of contracted development, clarify control issues such as restrictions on use of development equipment owned by contractors.
- Use tools that centrally manage and utilize configurations and settings, and foster technical skills to make full use of these tools in development teams.

S (4)-6	**Developer** Supplier Operator Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Technology: Arrangement of secure development environments Analyze risks throughout the SDLC, and protect and strengthen development-related environments.		
☐ **S(4)-6.1 Isolation and protection of environments** Isolate and protect the respective environments related to software development.		
Examples of measures	• Isolate the development and production environments. • Isolate environments and networks for software development (development, build, test, and distribution environments, etc.). • Minimize the use of production software and services from non-production environments in production environments. (SP800-218 PO.5.1 notional implementation example) • Regularly log, monitor, and audit trust relationships for authorization and access between environments and between components in the respective environments. (SP800-218 PO.5.1 notional implementation example) • Configure security controls and other tools related to the isolation and protection of environments to generate artifacts of environment behavior. (SP800-218 PO.5.1 notional implementation example) • Continuously monitor vulnerabilities of components deployed in the respective environments and implement risk-based measures by environment. • Configure and implement measures to protect hosting infrastructures of environments that comply with the Zero Trust Architecture.	

☐ **S(4)-6.2 Protection of development endpoints** Protect and strengthen endpoints designed for respective developers to perform development-related tasks using a risk-based approach.	
Examples of measures	• Select appropriate system protection methods (e.g., appropriate architecture, technology) based on risk analysis to isolate environments and networks for the purpose of software development. • Make security protection for development environments and endpoints (for software designers, developers, testers, builders, etc.) robust (using multi-factor authentication, risk-based authentication, conditional access by environment, encryption of sensitive data based on standards, etc.), monitor privileged access and access attempts, and detect, respond to, and restore cyber incidents. • Development environments should provide the minimum functionality required by users and services and must be configured following the principle of least privilege. • Strictly restrict connections to development environments (including limiting access to the Internet to the minimum necessary). • Implement hardening such as configuration management, change management, protection of development, maintenance environments, administrator privileges, etc. to prevent the creation and introduction of malicious software. (In the case of software for a system/service) • To streamline development and administrative work, prepare a common development platform including configuration management and provide it to the contractor (considering the burden of costs on business divisions).

Column	**Benefits of using AI in software development operation**

In 2023, a US survey found that 92% of US-based developers have been already using AI coding tools both at work and outside of work. This fact illustrates that generative AI is used extensively in software development operation.

In addition, in an analysis of GitHub Copilot users by a US research company, it was reported that, on average, developers accepted almost 30% of code suggestions from GitHub Copilot in the first year, and that the acceptance helped to improve productivity. Furthermore, it was found that the acceptance rate had increased as developers became more familiar with the tools. This suggests that it is likely to continue to affect the productivity of developers as they become more accustomed to software development with GitHub Copilot.

Another survey reported that less experienced developers benefit more from GitHub Copilot. Thus, it is hoped that generative AI will be used effectively in software development operation.

Column	**Negative aspects of AI use in software development operation**

A study by Stanford University in the US reported that giving too much authority to an AI assistant (for example, the automation of parameter selection) may reduce enthusiasm for addressing security vulnerabilities, and that AI assistants may reduce the proactiveness of developers in carefully searching library documentation for APIs and details of secure implementation.

Given that some causes of security vulnerabilities are related to the selection and use of inappropriate libraries, it is contemplated that developers need to pay attention to how they handle AI assistants (e.g., interactive methods including prompts), learn how to test the products they produce, etc.

Column	**Responses to ethical, legal, and social issues in AI use**

During the "Human Genome Project" in the 1980s, an initiative named Ethical, Legal and Social Implications (ELSI) was promoted. This initiative represents a way of thinking that addresses both technical issues as well as ethical, legal, and social influences, which is a perspective that should be emphasized in light of the rapidly advancing AI use at present. One of the ELSI initiatives in AI use that are being discussed is that for "trustworthy and responsible AI."

For example, the accuracy of a machine-learning model is greatly affected by both the quantity and quality (variation) of the training data, so it is important to consider characteristics related to the quality of data that affect performance and security, such as whether the data are biased, whether they can sufficiently predict events, and whether they contain noise. Before collecting such training data or utilizing machine-learning models, it is necessary to consider ethical (human rights violations, etc.), legal (copyright, unfair competition prevention, trade secrets, personal information and privacy, etc.), and social (AI fairness, transparency, accountability, etc.) impacts as risks and to establish an appropriate risk management system for the development and maintenance of "trustworthy and responsible AI."

In the EU, a provisional agreement was reached on a bill to regulate AI comprehensively (the Artificial Intelligence Act) as of December 2023. In the future, risk-based responses will be required for AI systems that are developed and used within the EU, and heavy fines will be set for violations; thus, each entity will be urged to take measures.

In addition, the US National Institute of Standards and Technology (NIST) has developed a framework (the NIST AI Risk Management Framework (RMF)) to better manage the risks associated with AI for individuals, organizations, and society, and the Trustworthy and Responsible AI Resource Center has started supporting its use.

https://airc.nist.gov/Home

Column **Example of secure software development practice for generative AI**

The US NIST has published SP800-218A (Secure Software Development Practices for Generative AI and Dual-Use Foundation Models Community Profile) as a derivative of the SSDF defined in SP800-218. This document supplements the SSDF with tasks, practices, and recommendations specific to AI model development and provides useful information for AI model developers, AI system developers, and AI system purchasers to gain a deeper understanding of secure software development techniques for AI models.

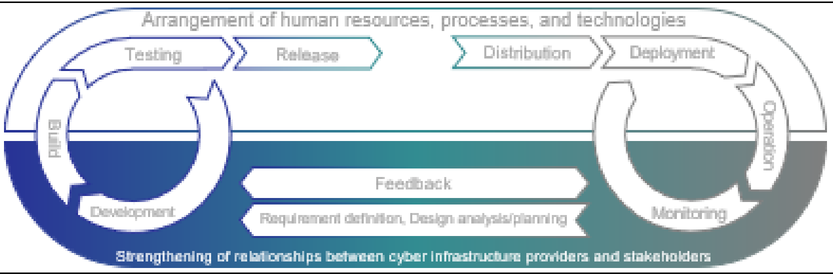
The SSDF has been supplemented with the following items:

- Data protection (added as the PS.1.2 task)
  Protect data for all training, testing, fine-tuning, and aligning from unauthorized access or modification.
- Model protection (added as the PS.1.3 task)
  Protect model weights and configuration parameters from unauthorized access or modification.
- SBOMs through supply chain levels of software artifacts (SLSA) (change in the PS.3.2 task)
  Collect, protect, maintain, and share provenance data for all components of the respective software releases (e.g., SBOMs through SLSA).
- Continuous monitoring of execution performance and behavior (added as the PO.5.3 task)
  Continuously monitor the execution performance and behavior of software in software development environments to identify potential suspicious activities or other issues.
- Analysis of data (added as the PW.3.1 task)
  Analyze data for signs of data poisoning, bias, homogeneity, and tampering before using them for the purposes of training, testing, fine-tuning, and aligning AI models and mitigate risks as necessary.

- Tracking of data provenance (added as the PW.3.2 task)
  Track the provenance of all training, testing, fine-tuning, and aligning data used for AI models.
- Adversarial samples (added as the PW.3.3 task)
  Include adversarial samples in training and test data to improve attack detection.

With reference to these tasks and practice examples, it is possible to establish systems, processes, and procedures for securely developing generative AI models.
https://csrc.nist.gov/pubs/sp/800/218/a/final

(5) Strengthening of relationships between cyber infrastructure providers and stakeholders

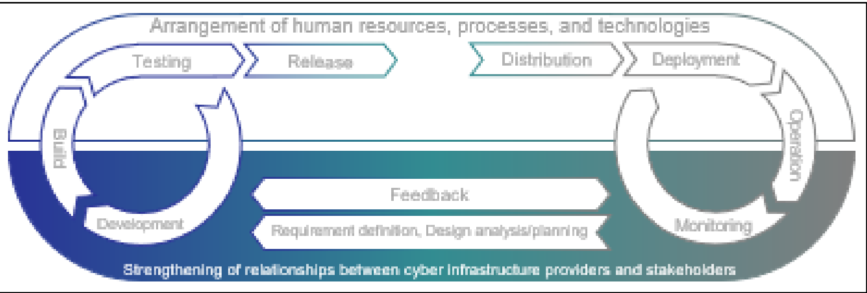
S (5)-1	**Developer**	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
	Supplier	
	Operator	
	Customer	
Organizational system for information sharing Establish an organizational structure for information sharing between private companies, relevant authorities, and specialized organizations to improve the security of software products and services.		
☐ **S(5)-1.1 Establishment of an organizational system for information sharing** Establish an organizational structure for information sharing between private companies, relevant authorities, and specialized organizations to improve the security of software products and services.		
Examples of measures	• Establish policies to check for and enforce compliance of in-house policies to legal requirements and industry best practices, and standards related to software. • Establish a CSIRT and a point of contact to make use of information linkage between private companies, relevant authorities, and specialized organizations regarding software security, and at the same time, advance the skills of those involved and promote the use of communication tools to improve efficiency.	
☐ **S(5)-1.2 Provision of important security-related information** Select and identify essential and important security-related information that is specific to the industry and provide it to partners in the supply chain.		
Examples of measures	• Establish a mechanism for information sharing on software security between cyber infrastructure providers (developers, suppliers, and operators) and customers (orderers). Suppliers contribute to information linkage as liaisons or intermediaries between developers and customers. • Actively share cases of damage (especially information on threats and countermeasures) to prevent the same damage from being repeated. (In the case of software for a system/service) • Establish a mechanism for sharing information between related vendors to improve the security of customers and provide responses when an incident occurs, based on a contract.	
☐ **S(5)-1.3 Use of vulnerability information notification services** Use vulnerability information notification services to share vulnerability information efficiently.		
Examples of measures	• For information sharing on the types of attacks and influences of a discovered vulnerability, make use of mechanisms operated by industry associations, etc. • For information sharing on vulnerabilities in settings, make use of institutions and mechanisms operated by specialized organizations, etc. • Promote the use of recommended information on common configuration management tools in the industry through an industry association.	

■ The need for a mechanism for sharing information among stakeholders

There is a growing need to formulate a mechanism for sharing information between specialized organizations, related vendors, or related parties, such as how contracts regarding responses when incidents occur, which will contribute toward improving customer security. Examples are provided as follows:

- At present, information sharing on vulnerabilities caused by software configurations is often limited to individual sharing between companies and personal connections. Therefore, sharing mechanisms such as a reporting system of public institutions (such as IPA) should be used.
- As it is generally difficult for developers to obtain information on the attacks and influences of a discovered vulnerability, it should be made possible to use the "Guidance for Sharing and Disclosure of Information on Damage from Cyberattacks" (established by The Study Group on Guidance for Sharing and Disclosure of Information on Damage from Cyberattacks on March 8, 2023), etc. ("Information sharing" refers to the exchange of technical information, mainly related to cyberattack techniques, conducted in private at information sharing venues or between specialized organizations. In contrast, "disclosure" is intended for victim organizations to present to the outside the status of the cyberattack damage that they suffered and details of their responses. Note that the guidance is expected to be more convenient by attaching importance and deadlines of measures.)
- As cyberattacks become more sophisticated and difficult for a single organization to clarify the full extent of an attack, it is important that "information sharing" is carried out promptly between other specialized organizations, not by a victim organization itself, but through specialized organizations that support victim organizations, from the perspective of preventing the spread of damage.
  In line with the recommendations made by the "Study Group for Promotion of Information Sharing on Damage Caused by Cyberattacks," a framework should be established for smooth information sharing between specialized organizations and its promotion by making use of the "Guide on How to Handle and Utilize Technical Information on Cyberattacks" and "Draft Model Contractual Articles on How to Handle Technical Information on Cyberattacks to be Included in NDA" (established by the study group on March 11, 2024).
- A public–private information sharing mechanism intended for all cyber infrastructure providers (developers, suppliers, and operators) and customers (orderers) should be established.

To establish such a mechanism, several challenges must be addressed, including assessment of eligibility for information sharing in information-sharing platforms, the arrangement of formats and manuals for effective use of information, etc. (Related requirement: S(5)-1.2)

S (5)-2	Developer Supplier Operator Customer	 Arrangement of human resources, processes, and technologies Testing Release Distribution Deployment Build Operation Feedback Development Requirement definition, Design analysis/planning Monitoring Strengthening of relationships between cyber infrastructure providers and stakeholders
Strengthening of cooperation systems To improve the security of software products and services, make use of systems and frameworks for cooperation with private companies, relevant authorities, and specialized organizations.		
☐ **S(5)-2.1 Utilization of cooperation systems** To improve the security of software products and services, make use of communities and cooperation systems aimed at improving software security, in which external businesses, customers, and specialized organizations participate.		
Examples of measures	• Participate in industry associations such as the Information Sharing and Analysis Center (ISAC) that share and analyze security information.	
☐ **S(5)-2.2 Contribution to cooperation systems** When participating in a community or cooperation system, actively participate in activities to contribute to the cooperation system.		
Examples of measures	• In addition to sharing damage information, make use of a wide range of cooperation frameworks, such as the following: ➢ Parent companies participate in a CSIRT council, and dispatch employees of group companies to the CSIRT s of the parent companies to share information. ➢ Share indicator of compromise (IoC) information within group companies by making use of the Malware Information Sharing Platform (MISP). ➢ Participate in the ISAC (Software ISAC, etc.) and CSIRT council. ➢ Engage in volunteer groups of the private sector and participate in conferences established by connecting government agencies and other organizations with a local security community and share vulnerability information with local businesses, business organizations, and local governments. ➢ Hold study sessions with contractors who do not have ongoing projects, and share information with them. ➢ User meetings hosted by prime providers, cross-industry briefings on sample cases of incidents and their causes. ➢ Utilization of government-established information sharing platform such as the IPA.	

■ Expectation for initiatives to strengthen cooperation systems among stakeholders

The initiatives described below are expected to strengthen cooperation systems formed to improve the security of products and services among stakeholders. (Related requirement: S(5)-2.2)

- Cooperation of industry associations is essential, and it is expected to contribute in various manners by encouraging active participation of the business sector in addressing challenges.
- It is expected that mechanisms for sharing information to deal with cyber threats beyond confidentiality obligations, contractual arrangements in response to an incident, means setting the level of information disclosure and its rules for information sharing in a supply chain, establishing frameworks to quickly report and share known vulnerabilities, and planning countermeasures to customers or establishing maintenance operation teams.
- As a means of supporting business operators to improve their level of security requirements, it is expected that guidelines and security baseline will be arranged and formulated by each industrial sector.

(6) Risk management by customers, and procurement and operation of secure software

S (6)-1	Developer	Supplier	Operator	Customer
Risk management under the leadership of the customer's management Integrate risk management that is implemented in cooperation with cyber infrastructure providers based on the leadership of the customer's management.				
☐ **S(6)-1.1 Risk management** Implement risk management in which the customer's independent and proactive efforts are integrated with efforts based on a contract with cyber infrastructure providers.				
Examples of measures	• Customers are responsible for risk management of the entire systems they own. Appropriate measures should be implemented based on the potential risks of the entire system (such as the application of multi-factor authentication to users with administrative privileges and realization of efficiency through proper operation of single sign-on). • Regard critical cyber infrastructure providers that support the organizational security posture as critical business functions, and provide funding for entire life cycles of operation and risk responses of the intended systems and software according to their importance to organizational success, considering confirmed results of checks of proposals from cyber infrastructure providers and cost breakdowns. • Also refer to the assumed case examples that do not pose an issue under the Antimonopoly Act or the Act on Ensuring Proper Transactions Involving Specified Entrusted Business Operators in terms of security measures, applicable to both the ordering party and the counterparty, along with their explanatory documents. They supplement “Toward building partnerships with suppliers to enhance cybersecurity across the supply chain,” a document prepared by the Ministry of Economy, Trade and Industry and the Japan Fair Trade Commission.[15] • Require cyber infrastructure providers to have transparency in their position on internal control and roadmap for following secure by design and secure by default practices. • Assuming a case in which an incident occurs in the operation of a system owned by a customer, who enters into a maintenance contract with the cyber infrastructure provider to which the maintenance of the system is entrusted, including incident response and allocation of roles for it.. • Create a plan to improve the capabilities of cyber infrastructure providers that follow secure by design and secure by default practices. • Define all roles and responsibilities involved in the software operation life cycle, including cybersecurity staff, security champions, security testers, operation and platform engineers, and procurement staff. • When using cloud systems, clarify security responsibilities of customers and suppliers based on a shared responsibility model, and prioritize cloud providers with high transparency in their security position, internal controls, and ability to fulfill responsibilities under the shared responsibility model.			

[15] https://www.meti.go.jp/press/2025/12/20251226001/20251226001-e.pdf

☐ **S(6)-1.2**	**Resource arrangement** Allocate and develop resources to respond proactively to known vulnerabilities and implement mitigation measures (including SBOM utilization).
Examples of measures	• Check support periods of software products and create an operation plan that does not use out-of-support software. • Request and verify evidence information related to security implementation of software products (such as self-conformance certificates that prove the conformity to recommended SBOM/SSDF implementation practices). • Perform integrity mechanism checks, security tests, environmental tests, and functional tests before software acceptance or deployment. • To ensure the quality of the software to be introduced, establish quality verification procedures and standards through discussions between the customer and cyber infrastructure provider and request evidence. • Continuously conduct security monitoring of the introduced software, and report to the cyber infrastructure provider when a suspected software vulnerability is identified. • Determine an update policy based on software update strategies and adopt an automated update mechanism as necessary.
☐ **S(6)-1.3**	**Utilization of collaborative systems** Proactively participate in and utilize communities and collaborative systems aimed at improving software security.
Examples of measures	• When participating in a community or cooperation system, actively participate in activities to contribute to the cooperation system. Examples are provided below. ➢ Participation in ISAC (e.g., Software ISAC) and CSIRT council ➢ Collaboration through communities composed of volunteer groups formed by private companies, and regional security communities

■ Differences in life cycles

Life cycles (usage periods) that customers who use software recognize are different from life cycles (support periods) that cyber infrastructure providers who provide the software recognize. When using software, it is essential to regularly check support periods of versions of software to be used and create an operation plan for using the software for which the support period has expired.

(Related requirement: S(6)-1.2)

S (6)-2	Developer	Supplier	Operator	Customer
Software procurement/operation under the leadership of the customer's management Procure and operate software securely under the leadership of the customer's management.				

☐ **S(6)-2.1 Definition of security requirements** Define security requirements for incorporating security functions into software design plans and present them to cyber infrastructure providers before procuring and deploying software.	
Examples of measures	• Work with industry counterparts to request that cyber infrastructure providers prioritize secure by design and secure by default initiatives in the future.
☐ **S(6)-2.2 Disclosure of security practice requirements** Disclose security practice requirements for cyber infrastructure providers before procuring and deploying software.	
Examples of measures	• Give permissions to IT departments and information security departments to specify purchasing criteria that emphasize secure by design and secure by default practices.
☐ **S(6)-2.3 Decision-making based on risk assessment** When procuring and introducing software, make decisions based on risk assessment.	
Examples of measures	• Create a policy that requires IT departments and information security departments to evaluate the security of software before purchasing it and to ask for necessary information sources, and authorize the departments to object to the purchase of software that does not meet the standards. • When making a decision to accept risks related to a specific technology product, create formal documentation and have senior management give approval and regularly make a report to the board of directors. • When introducing a digital service, evaluate a migration possibility to other digital services from a risk perspective and make appropriate decision on introduction.
☐ **S(6)-2.4 Budget securement** Continuously secure budgets related to introduction, operation, migration, disposal, risk response, and related contracts, considering software life cycles.	
Examples of measures	• Consider measures to reduce remaining cybersecurity risks below an acceptable level, secure resources (budget, personnel, etc.) required for the implementation, and then work on specific measures. (Cybersecurity Management Guidelines v3, Direction 3) • Anticipate such risks as the discovery of vulnerabilities in OSS or third-party components during development or operation, and budget costs accordingly. (In the case of software for a system/service) • As a prerequisite for promoting the sharing of cost recognition among stakeholders, understand that leaving vulnerabilities unsolved will lead to future liabilities (damage from cyberattacks, etc.) as a common management risk.

5.5. Relationship between the Common Standards and Guidelines

(1) Framework for the use of the Common Standards and its positioning

National administrative agencies and independent administrative agencies (hereinafter referred to as "government agencies, etc.") are to ensure information security within their respective organizations in accordance with the framework for the use of the "Common Standards for Cybersecurity Measures for Government Agencies and Related Agencies"[16] (hereinafter referred to as "Common Standards") published by National center of Incident readiness and Strategy for Cybersecurity (NISC, current National Cybersecurity Office (NCO)).

Within this framework, to comply with common norms and standards, which are requirements for their implementation, government agencies and other institutions are required to formulate information security policies based on the characteristics of their organizations and the information that they handle, while referring to the "Guidelines for Formulating Measures Criteria for Government Agencies and Related Agencies" (hereinafter referred to as the "Guidelines for Formulating Measures Criteria") to establish operational regulations and implementation procedures related to the countermeasures set out in the policies, and to implement countermeasures in a planned manner.

“Common Standards” consist of a common model, common standards, and guidelines for formulating the measurement criteria. Common Standards classify the measures that government agencies should implement into three hierarchical levels—division, section, and subsection—based on their objectives and outline the purpose, intent, and items to observe at the third level (subsection). The Guidelines for Formulating Measures Criteria provide examples of basic countermeasures that should be implemented to meet the criteria and approaches for the formulation and implementation of information security policies.

(2) Relationship with software handled by the Common Standards

"Software" covered by the Guidelines refers to the following types of software handled by cyber infrastructure providers, in accordance with the purpose of the Guidelines to "promote effective cybersecurity measures intended for software in supply chains." (For details, see "1.3. Applicable objects: (1) Scope of software" in the Guidelines.)

- Software product
- Software service
- Embedded software
- Software that constitutes a system or service

In addition, the scope of software for which the Common Standards require enhanced measures in the procurement of external contractors and the outsourcing of development and operation of information systems is as follows. It is assumed that the scope matches the software for which the Guidelines are intended
<Outsourcing (procurement)>

- Cloud service
- Equipment (server equipment, terminal, communication line equipment, multifunction printer, specific purpose equipment, software, etc.)
  *In terms of software that is deemed particularly necessary to address supply chain risks as <an example of software that manages or controls the foundation of an information system>, the following are listed as examples:
  - ➢ Software that controls terminals, server equipment, and communication line equipment.

[16] https://www.nisc.go.jp/policy/group/general/kijun.html
The Cybersecurity Strategy Headquarters determines "common norms" and "common standards" of the Common Standards.

    - Software that manages comprehensive entity authentication
    - Software that controls and manages networks
    - Software that manages assets
    - Software related to monitoring
    - Software used as a security function of an information system

<Information system (outsourcing of development and operation)>

- Application contents

In addition, reinforcement-related software security and supply chain risk measures have been actively implemented in the most recent revision of the Common Standards. The following items are listed as key points in the revision of the Common Standards (2023 edition). Thus, they agree with the purpose of the Guidelines:

- Key points in the revision of the Common Standards (2023 edition)[17]
    - Strengthening of supply chain measures related to information security
    - Strengthening of measures in light of the expanding use of cloud services
    - Strengthening of measures for software use
    - Strengthening of measures in light of strengthening of cyber resilience, cyber threat and technology trends
    - Strengthening of cross-organizational information security measures and assurance of measures according to the importance of information systems

(3) Relationship between the Common Standards and Guidelines

The Guidelines specify the appropriate division of roles and responsibilities between cyber infrastructure providers and customers to ensure the cyber security of software and improve resilience. In terms of the relationship with the Common Standards, government agencies are the customers and they comply with and reference these standards. Conversely, cyber infrastructure providers are external contractors and are positioned to accept outsourced business—such as software development, system operations, or the supply of equipment and related services. The Common Standards do not directly state the roles and responsibilities that cyber infrastructure providers should fulfill as contractors to enable customers to implement items to observe; therefore, it is necessary to read the contents of the Common Standards from the perspective of a contractor to understand it.

The first level of the Common Standards is divided into seven parts (Parts 2 to 8). Some parts contain content that is directly related to the responsibilities set forth in the Guidelines (items marked with "○" in the table below) and some parts contain descriptions related to the requirements as responsibilities set forth in the Guidelines (items marked with "△" in the table below).

Table 8 Correspondence relationship with the first layer (chapter) of the Common Standards

First layer (chapter) of the Common Standards	Cyber infrastructure provider	Customer
Chapter 1: General Provisions		
Chapter 2: Basic Framework of Information Security Measures		○
Chapter 3: Information Handling		△

17 https://www.nisc.go.jp/pdf/policy/general/rev_pointr5.pdf

First layer (chapter) of the Common Standards	Cyber infrastructure provider	Customer
Chapter 4: Outsourcing	○	○
Chapter 5: Life of Information Systems	○	○
Chapter 6: Information Systems Components	○	○
Chapter 7: Security Requirements for Information Systems	○	○
Chapter 8: Use of Information Systems	△	△
Appendix		

In addition, the classification of the chapters in the second layer presents a relationship between the Common Standards and Guidelines; as the following chapters of the Common Standards (items in a red frame in the table below) are both marked "○" for customers and cyber infrastructure providers, there is a particularly strong relationship with the responsibilities and division of roles shown in the Guidelines.

4.1 Subcontracting
4.2 Use of Cloud Services
4.3 Procurement of Equipment, etc.
5.2 Measures at Each Phase of Information System Lifecycle
6.4 Communication Lines
6.5 Software
6.6 Applications and Content
7.2 Measures against Information Security Threats

Table 9 Correspondence relationship with the second layer (chapter) of the Common Standards

Second layer (chapter) of the Common Standards		Cyber infrastructure provider	Customer
Chapter 1: General Provisions			
	1.1 Purpose and Scope of these Common Standards for Measures		
	1.2 Classification of Information and Handling Restrictions		
	1.3 Definition of Terms		
	1.4 Terminology		
	1.5 Basic Measures and Explanations		
Chapter 2: Basic Framework of Information Security Measures			○
	2.1 Introduction and Plan		○
	2.2 Operation		○
	2.3 Assessment		△
	2.4 Review		△
	2.5 Incorporated Administrative Agencies and Designated Corporations		△
Chapter 3: Information Handling			△
	3.1 Information Handling		△
	3.2 Information Handling Areas		△
Chapter 4: Outsourcing		○	○
	4.1 Subcontracting	○	○

	4.2 Use of Cloud Services	○	○
	4.3 Procurement of Equipment, etc.	○	○
Chapter 5: Life of Information Systems		○	○
	5.1 Classification of Information Systems		△
	5.2 Measures at Each Phase of Information System Lifecycle	○	○
	5.3 Operational Continuity Plan of Information Systems	△	△
	5.4 Shared Government Systems	△	△
Chapter 6: Information Systems Components		○	○
	6.1 Terminals	△	△
	6.2 Server Equipment	△	△
	6.3 Multifunction Devices and Equipment for Specific Purposes	△	△
	6.4 Communication Lines	○	○
	6.5 Software	○	○
	6.6 Applications and Content	○	○
Chapter 7: Security Requirements for Information Systems		○	○
	7.1 Security Functions of Information Systems	△	△
	7.2 Measures against Information Security Threats	○	○
	7.3 Zero Trust Architecture	△	△
Chapter 8 Use of Information Systems		△	△
	8.1 Use of Information Systems	△	△
Appendix			

5.6. Relationship between the Guidelines for Establishing Safety Principles for Ensuring Information Security of Critical Infrastructure and Guidelines

(1) Framework for the use of the Guidelines for Establishing Safety Principles for Ensuring Information Security of Critical Infrastructure and its positioning

Critical infrastructure operators conduct their business in accordance with the relevant standards specified under the legal system related to their business fields. In the "Action Plan for Cybersecurity of Critical Infrastructure" (hereinafter referred to as the "Action Plan"[18]), it is stated that critical infrastructure operators shall endeavor to strengthen their own organizational failure response systems based on the safety standards (described later), and through these efforts, information security measures related to critical infrastructures are being advanced comprehensively. In the "Guidelines for Establishing Safety Principles for Ensuring Information Security of Critical Infrastructure"[19] (hereinafter referred to as the "Guidelines for Establishing Safety Principles"), with respect to cybersecurity assurance, these standards or reference documents regarding the judgments and actions of respective critical infrastructure operators are defined as "safety standards"[20], and efforts to ensure cybersecurity that are commonly required for each critical infrastructure field are classified and organized. It is expected that these efforts will be described in the safety standards to be developed by critical infrastructure industry in principle.

In addition, a manual has been developed as a reference document in the Guidelines for Establishing Safety Principles, which explains the basic approaches and specific procedures for promoting security measures, such as risk management.

(2) Relationship with software handled by the Guidelines for Establishing Safety Principles

The "software" covered by the Guidelines refers to the following types of software handled by cyber infrastructure providers, in accordance with the purpose of the Guidelines to "promote effective cybersecurity measures intended for software in supply chains." (For details, see "1.3. Applicable objects: (1) Scope of software" in the Guidelines .)

- Software product
- Software service
- Embedded software
- Software that constitutes a system or service

In addition, the intended scope to be stipulated in the safety standards through the Guidelines for Establishing Safety Principles should be based on the contents of "Examples of Targeted Critical Systems" described in "Appendix 1: Examples of Intended Critical Infrastructure Businesses, etc. and Critical Systems" of the Action Plan and "Critical Infrastructure Services (including Procedures)," "Examples of Critical Infrastructure Service Disruptions," and "Service Maintenance Levels" described in "Appendix 2: Critical Infrastructure Services and Service Maintenance Levels." As an example of a supply chain that should be addressed, a cloud service is shown, whereas information systems, control systems, and general-purpose equipment are shown as those that are subject to risk management. Based on the fact that software is an element that constitutes a system, the following are assumed as target software. Therefore, it is assumed that, of the software that the Guidelines target, those other than software products comply.

[18] https://www.nisc.go.jp/pdf/policy/infra/cip_policy_2024.pdf
[19] https://www.nisc.go.jp/pdf/policy/infra/shishin202307.pdf
[20] Classified as "internal regulations" that critical infrastructure operators establish for themselves to meet the expectations of the public and users, etc. and cross-industry "industry standards" and "guidelines" that industry associations, etc. establish to satisfy "mandatory standards" established by the government based on relevant laws and regulations, "recommended standards" and "guidelines" established by the government in accordance with relevant laws and regulations, and the expectations of relevant laws and regulations and the public.

<Outsourcing (procurement)>

- Cloud service
- Control system (including general-purpose equipment)

<Information system (outsourcing of development and operation)>

- Information system

The Guidelines for Establishing Safety Principles state the following, which agrees with the purpose of the Guidelines:

4.4. Supply-chain Risk Management
Understand dependencies between organizational critical systems and functions and supply chains, and understand the status of security measures of suppliers. Conduct risk assessments and risk responses for supply chain risks. (Omitted) For tier 1 suppliers, clarify roles and scope of responsibilities to be assumed in response to cybersecurity risks in contracts between businesses. Furthermore, it is desirable to implement risk management for entire supply chains by ensuring that respective suppliers are aware of the implementation status of supply chain risk management within their lower-tier suppliers, while determining degrees of involvement with Tier 2 suppliers according to the type of risk. In addition, it is desirable to increase the effectiveness of measures throughout supply chains with support from suppliers for introduction of security measures, joint implementation, etc.

(3) Relationship between the Guidelines for Establishing Safety Principles and Guidelines

The Guidelines specify the appropriate division of roles and responsibilities between cyber infrastructure providers and customers to ensure the cyber security of software and improve resilience. In relation to the Guidelines for Establishing Safety Principles, customers are considered as critical infrastructure operators—entities that implement measures on the basis of safety standards—based on the Guidelines for Establishing Safety Principles, whereas cyber infrastructure providers are external contractors[21] from the perspective of critical infrastructure operators and are usually in a position to accept business outsourcing, including software, from critical infrastructure operators (customers)—entities for the development/operation of information systems or businesses from which equipment is procured. The Guidelines for Establishing Safety Principles do not directly state the roles and responsibilities that cyber infrastructure providers should fulfill as contractors to enable customers to realize efforts toward ensuring cybersecurity common to respective critical infrastructure fields; therefore, it is necessary to read the content of the Guidelines for Establishing Safety Principles from the perspective of a contractor in order to understand it.

The Guidelines for Establishing Safety Principles are classified based on the efforts required for ensuring cybersecurity, which are common to respective critical infrastructure fields. Some chapters contain content that is directly related to the responsibilities set forth in the Guidelines (items marked with "○" in the table below), whereas some chapters contain descriptions related to the requirements as responsibilities set forth in the Guidelines (items marked with "△" in the table below).

[21] However, if critical infrastructure operators conduct development and supply in-house, it is considered that they will take on the responsibilities of "developer" and "supplier," which are the respective roles they shall take as a "cyber infrastructure provider" in the category of responsibilities, as the "(Entity)." For specific examples, see Table 4 e in "1.4 Approach to the division of roles."

Table 10 Correspondence relationship with the first layer of the Guidelines for Establishing Safety Principles

First level of the Guidelines for Establishing Safety Principles	Cyber infrastructure provider	Customer
1. Purposes and Positioning		
2. General Provisions		△
3. Cybersecurity in Organizational Governance		△
4. Utilization of Risk Management, and Crisis Management	○	○
5. Measures	○	○

In the classification of chapters in the second layer, in the relationship between the Guidelines for Establishing Safety Principles and the Guidelines, because the following chapters of the Guidelines for Establishing Safety Principles (items in a red frame in the table below) are marked "○" for both customers and cyber infrastructure providers, there is a particularly strong relationship with the responsibilities and division of roles shown in the Guidelines.

4.2. Risk Management
4.3. Addressing Cybersecurity Risks
4.4. Supply-chain Risk Management
4.8. Operation During Normal Times
5.1 Organizational Measures

Table 11 Correspondence relationship with the second layer of the Guidelines for Establishing Safety Principles

Second level of the Guidelines for Establishing Safety Principles		Cyber infrastructure provider	Customer
1. Purposes and Positioning			
	1.1. The Importance of Ensuring Cybersecurity for Critical Infrastructure (CI)		
	1.2. What are "Safety Principles"?		
	1.3. Positioning of the Guideline for Establishing Safety Principles		
2. General Provisions			△
	2.1. Purpose of Formulating the Safety Principles		
	2.2. Applicable Scope		
	2.3. Roles of Stakeholders		△
3. Cybersecurity in Organizational Governance			△
	3.1. Organizational Policy		△
	3.2. Communication Within and Outside the Organization		△
	3.3. Managing Cybersecurity Risks as Business Risks		○
	3.4. Assignment of Responsibilities and Authority		△
	3.5. Securing Resources		△
	3.6. Auditing and Monitoring		△
	3.7. Information Disclosure		

Second level of the Guidelines for Establishing Safety Principles		Cyber infrastructure provider	Customer
	3.8. Continuous Improvement		△
4. Utilization of Risk Management, and Crisis Management		◯	◯
	4.1. Understanding the Organization’s Situation	△	△
	4.2. Risk Management	◯	◯
	4.3. Addressing Cybersecurity Risks	◯	◯
	4.4. Supply-chain Risk Management	◯	◯
	4.5. Business Continuity Plan and Other Plans		
	4.6. Human Resource Development and Awareness-Raising		△
	4.7. Establishment of CSIRT, etc.	△	△
	4.8. Operation During Normal Times	◯	◯
	4.9. Crisis Management	△	△
	4.10. Exercises and Training	△	△
5. Measures		◯	◯
	5.1 Organizational Measures	◯	◯
	5.2 Personnel Measures	△	△
	5.3 Physical Measures		
	5.4 Technical Measures	△	△
	5.5. Measures Based on Trends	△	△

5.7. Reference information

(1) List of reference information

Abbrev.	Document title
NSA	SECURING THE SOFTWARE SUPPLY CHAIN / Recommended Practices Guide for Developers https://media.defense.gov/2022/Sep/01/2003068942/-1/-1/0/ESF_SECURING_THE_SOFTWARE_SUPPLY_CHAIN_DEVELOPERS.PDF
NSA-S	SECURING THE SOFTWARE SUPPLY CHAIN / Recommended Practices Guide for Suppliers https://media.defense.gov/2022/Oct/31/2003105368/-1/-1/0/SECURING_THE_SOFTWARE_SUPPLY_CHAIN_SUPPLIERS.PDF
NSA-C	SECURING THE SOFTWARE SUPPLY CHAIN / Recommended Practices Guide for Customers https://media.defense.gov/2022/Nov/17/2003116445/-1/-1/0/ESF_SECURING_THE_SOFTWARE_SUPPLY_CHAIN_CUSTOMER.PDF
SP800-218	NIST SP800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf
BSA	The BSA Framework for Secure Software: A New Approach to Securing the Software Lifecycle https://www.bsa.org/files/reports/bsa_software_security_framework_web_final.pdf
CISA-D	Defending Against Software Supply Chain Attacks https://www.cisa.gov/sites/default/files/publications/defending_against_software_supply_chain_attacks_508_1.pdf
CISA-SBD	Secure-by-Design - Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software https://www.cisa.gov/sites/default/files/2023-10/SecureByDesign_1025_508c.pdf
SP800-161	NIST SP800-161 Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-161r1-upd1.pdf
ISMS	ISO/IEC 27002:2022 - Information security, cybersecurity and privacy protection Information security controls https://www.iso.org/standard/75652.html

Abbrev.	Document title
ISO 15408	Common Criteria for Information Technology Security Evaluation ISO/IEC 15408:2022 - Information security, cybersecurity and privacy protection Evaluation criteria for IT security Part1～3 https://www.iso.org/standard/72891.html https://www.iso.org/standard/72892.html https://www.iso.org/standard/72906.html
DSP	ENISA Guidelines on assessing DSP and OES compliance to the NISD security requirements https://op.europa.eu/en/publication-detail/-/publication/78f2a620-f909-11e8-9982-01aa75ed71a1/language-en
Ministry of Internal Affairs and Communications	Guidelines for Information Security Measures in Cloud Service Provision https://www.soumu.go.jp/main_content/000771515.pdf Guidelines for Information Disclosure regarding the Safety and Reliability of Cloud Services https://www.soumu.go.jp/main_content/000477838.pdf
CRA	The European Cyber Resilience Act https://www.europarl.europa.eu/RegData/etudes/BRIE/2022/739259/EPRS_BRI(2022)739259_EN.pdf REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on horizontal cybersecurity requirements for products with digital elements and amending Regulation (EU) 2019/1020 https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ:L_202402847
Common Standards	Common Standards for Cybersecurity Measures for Government Agencies and Related Agencies https://www.nisc.go.jp/pdf/policy/general/kijyunr5.pdf Common Model for Cybersecurity Measures for Government Agencies and Related Agencies https://www.nisc.go.jp/pdf/policy/general/kihanr5.pdf Guidelines for Formulating Measures Criteria for Government Agencies and Related Agencies https://www.nisc.go.jp/pdf/policy/general/guider6.pdf
Guidelines for Establishing Safety Principles	Action Plan for Cybersecurity of Critical Infrastructure https://www.nisc.go.jp/pdf/policy/infra/cip_policy_abst_2024.pdf Guidelines for Establishing Safety Principles for Ensuring Information Security of Critical Infrastructure https://www.nisc.go.jp/pdf/policy/infra/shishin5.pdf
Japan-US-Australia-India Cybersecurity Partnership	Joint Statement of the Japan-US-Australia-India Summit (QUAD Joint Principles) https://www.mofa.go.jp/mofaj/fp/nsp/page1_001188.html
UN-R155	UN Regulation No. 155 - Cyber security and cyber security management system https://unece.org/sites/default/files/2023-02/R155e%20%282%29.pdf
ISO 21434	ISO/SAE 21434:2021 - Road vehicles Cybersecurity engineering https://www.iso.org/standard/70918.html
UN-R156	UN Regulation No. 156 - Software update and software update management system https://unece.org/sites/default/files/2024-03/R156e%20%282%29.pdf

Abbrev.	Document title
ISO 24089	ISO 24089:2023 - Road vehicles Software update engineering https://www.iso.org/standard/77796.html ISO 24089:2023/Amd 1:2024 https://www.iso.org/standard/87522.html
OMB M-23-16	Update to Memorandum M-22-18, Enhancing the Security of the Software Supply Chain through Secure Software Development Practices https://bidenwhitehouse.archives.gov/wp-content/uploads/2023/06/M-23-16-Update-to-M-22-18-Enhancing-Software-Security.pdf Secure Software Development Attestation Form Instructions https://www.cisa.gov/sites/default/files/2024-03/Self-Attestation-Common-Form-03082024-FINAL.pdf
SP800-218A	NIST SP 800-218A Secure Software Development Practices for Generative AI and Dual-Use Foundation Models An SSDF Community Profile https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218A.pdf
DSIT	DSIT – The Code of Practice for Software Vendors https://www.gov.uk/government/calls-for-evidence/a-code-of-practice-for-software-vendors-call-for-views/call-for-views-on-the-code-of-practice-for-software-vendors
Cybersecurity Management Guidelines	Cybersecurity Management Guidelines https://www.meti.go.jp/policy/netsecurity/downloadfiles/guide_v3.0.pdf

(2) Relationships with other standards, guidelines, etc.

The Guidelines have a relationship as described below with various other guidelines and frameworks intended for software security and software development assurance. It is possible to use these guidelines in specific initiatives, when further policy considerations and means of implementation are required. The scope on which the Guidelines are based and their relationships with other major standards and guidelines are shown in Figure 8. The relationships with major standards, guidelines, etc. are described below.

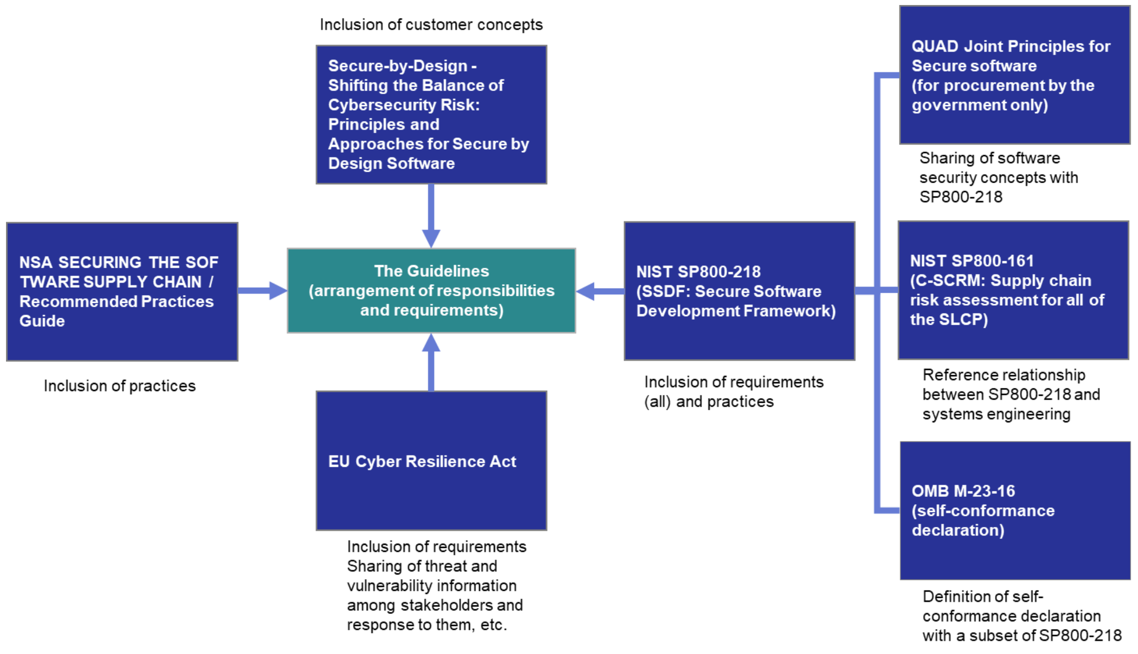

Inclusion of customer concepts
Secure-by-Design - Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software
QUAD Joint Principles for Secure software (for procurement by the government only)
Sharing of software security concepts with SP800-218
NSA SECURING THE SOF TWARE SUPPLY CHAIN / Recommended Practices Guide
The Guidelines (arrangement of responsibilities and requirements)
NIST SP800-218 (SSDF: Secure Software Development Framework)
NIST SP800-161 (C-SCRM: Supply chain risk assessment for all of the SLCP)
Inclusion of practices
Inclusion of requirements (all) and practices
Reference relationship between SP800-218 and systems engineering
EU Cyber Resilience Act
OMB M-23-16 (self-conformance declaration)
Inclusion of requirements Sharing of threat and vulnerability information among stakeholders and response to them, etc.
Definition of self-conformance declaration with a subset of SP800-218

Figure 8 Relationship between the Guidelines and other standards, guidelines, etc.

[1] NIST SP800-218

SP800-218 issued by NIST provides guidance for strengthening the security of software supply chains. It advocates a framework known as the SSDF, which adds secure software practices to SDLC models to ensure security of the software under development. The Guidelines comprehensively treat the tasks required to execute each practice shown in SP800-218 as "requirements," and use implementation examples for the respective tasks as reference for some "Examples of measures."

[2] NSA Software Supply Chain Guidance (for Developers, Suppliers, Customers)

The three editions of the software supply chain guidance issued by the NSA (for developers, suppliers, and customers) provide industry best practices and principles to which software developers, suppliers, and customers should refer. The principles outlined in the Developer Edition include planning security requirements, designing the software architecture from a security perspective, implementing security functions, and maintaining the security of the software development infrastructure (development environment, source code review, testing, etc.). The Guidelines mainly reference some "Examples of measures" based on the essence of the principles and best practices described in the three documents.

[3] CISA Secure-by-Design - Shifting the Balance of Cybersecurity Risk: Principles and Approaches for Secure by Design Software

The guidance published by the CISA on the principles and approaches of secure by design is intended to ask software developers to prioritize security when designing, developing, and offering products. It advocates three principles, namely "Having ownership of customer security outcomes," "Accepting fundamental transparency and accountability," and "Leading from the top," and explains the respective principles, key practices, and tactics (techniques) from the perspectives of secure by design and secure by default. The Guidelines constitute requirements for responsibilities in accordance with the principles of secure by design and secure by default, and presents the essence of the security principles of software products and the tactics (techniques) as a reference in the "Examples of measures."

[4] EU CRA

The CRA is a legal framework that provides cybersecurity requirements for hardware and software products with digital elements within the EU. It is expected to be fully enforced in 2027. This legal framework covers a wide range of products with digital elements, with some exceptions, and manufacturers who deploy products to the European market will be obligated to ensure that they meet security requirements throughout their product life cycles (such as creating SBOMs, providing security updates, and reporting to authorities when vulnerabilities are discovered or in the event of an incident). In the Guidelines, cybersecurity requirements (requirements relating to product characteristics and vulnerability handling requirements) and some information and instructions to users are used as a reference for "Itemized requirements" or "Examples of measures."

[5] Other standards and guidelines

In addition to those mentioned above, the Guidelines also refer to the following standards and guidelines:

- The BSA Framework for Secure Software: A New Approach to Securing the Software Lifecycle
- CISA Defending Against Software Supply Chain Attacks
- NIST SP800-161 Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations
- ENISA Guidelines on assessing DSP and OES compliance to the NISD security requirements
- Common Criteria for Information Technology Security Evaluation (ISO/IEC 15408)
- ISO/IEC 27002:2022
- Ministry of Internal Affairs and Communications Guidelines for Information Security Measures in Cloud Service Provision
- Ministry of Internal Affairs and Communications Guidelines for Information Disclosure regarding the Safety and Reliability of Cloud Services
- National Cybersecurity Office Common Standards for Cybersecurity Measures for Government Agencies and Related Agencies
- National Cybersecurity Office Guidelines for Formulating Measures Criteria for Government Agencies and Related Agencies
- National Cybersecurity Office Action Plan for Cybersecurity of Critical Infrastructure

- National Cybersecurity Office Guidelines for Establishing Safety Principles for Ensuring Information Security of Critical Infrastructure
- Japan-US-Australia-India Cybersecurity Partnership Joint Statement of the Japan-US-Australia-India Summit (QUAD Joint Principles)
- UN-R155
- UN-R156
- ISO/SAE 21434:2021
- ISO 24089:2023
- OMB M-23-16
- NIST SP800-218A
- DSIT The Code of Practice for Software Vendors

Together with them, the software-related C-SCRM framework and recommendations for identifying, evaluating, and mitigating risk based on SSDFs published by the CISA are used as references.

(3) Correspondence relationships with NIST SP800-218

Requirements	Corresponding items in NIST SP800-218
S(1)-1	PW.1.1, PW.1.2, PW.2.1
S(1)-2	PW.5.1, PW.6.1, PW.6.2, PW.7.1, PW.7.2, PW.9.2
S(1)-3	PW.8.1, PW.8.2, PW.9.1
S(1)-4	PW.9.1, (PS.1.1, PO.5.1, PO.5.2)
S(2)-1	PW.1.3, PW.4.1, PW.4.2, PW.4.4, (RV.2.1)
S(2)-2	PS.1.1, PS.3.1, PS.3.2
S(2)-3	PO.1.3, (PW.4.4)
S(2)-4	PS.2.1, PW.9.2
S(3)-1	RV.1.1, RV.1.2, RV.1.3
S(3)-2	RV.2.1, RV.2.2
S(3)-3	RV.3.1, RV.3.2, RV.3.3, RV.3.4, PW.7.2
S(4)-1	PO.2.1, PO.2.2, PO.2.3, (PO.3.1, PO.3.2, PO.3.3)
S(4)-2	PO.1.1, PO.1.2, (PO.2.3)
S(4)-3	(PO.1.1, PO.1.2, PO.2.3)
S(4)-4	PO.4.1, PO.4.2
S(4)-5	PO.3.1, PO.3.2, PO.3.3
S(4)-6	PO.5.1, PO.5.2
S(5)-1	—
S(5)-2	—
S(6)-1	—
S(6)-2	—

(4) Correspondence relationship between the three NSA Software Supply Chain Guidance documents

Requirements	NSA (for Developers)	NSA-S (for Suppliers)	NSA-C (for Customers)
S(1)-1	2.3.2	2.3.1	—
S(1)-2	2.2.1.4, 2.2.2, 2.2.6, 2.2.3.2, 2.3.2, 2.3.3, 2.4.1	2.2.2, 2.3.3, 2.3.4, 2.3.6	—
S(1)-3	2.2.1.3, 2.2.3.2, 2.3.2, 2.4.1	2.2.2, 2.3.5, 2.3.6	—
S(1)-4	—	—	—
S(2)-1	2.2.3, 2.3.2, 2.3.3, .2.3.4, 2.3.5	2.3.1, 2.3.2	2.1, 2.2
S(2)-2	2.2.1.1, 2.2.1.2, 2.2.1.4, 2.2.6, 2.3.2, 2.3.3, 2.4.1, 2.5.3	2.2.1, 2.2.2, 2.2.3	—
S(2)-3	2.2.3	2.1.1	—
S(2)-4	—	—	—
S(3)-1	2.3.4, 2.4.1	2.4.1	—
S(3)-2	—	—	—
S(3)-3	—	—	—
S(4)-1	—	—	—
S(4)-2	2.2.3	2.1.1	—
S(4)-3	2.2.3	2.1.1	—
S(4)-4	—	—	—
S(4)-5	—	—	—
S(4)-6	—	—	—
S(5)-1	—	—	—
S(5)-2	—	—	—
S(6)-1	—	—	2.1, 2.2, 2.3
S(6)-2	—	—	2.1

(5) Correspondence relationship with CISA Secure-by-Design - Shifting the Balance of Cybersecurity Risk

Requirements	Principle 1	Principle 2	Principle 3	Secure by design tactics	Secure by default tactics
S(1)-1	[SBD]-1 [SPD]-5	—	—	11, 12	—
S(1)-2	[SBD]-5	—	—	1, 4, 5, 6, 7	1
S(1)-3	[SBD]-2	—	—	6	—
S(1)-4	—	—	—	—	4
S(2)-1	[PSB]-3,4 [SPD]-4	—	—	3	—
S(2)-2	—	[SPD]-5	—	8	—
S(2)-3	—	—	—	—	—
S(2)-4	[SBD]-1,4	—	—	—	1, 5, 8
S(3)-1	—	[SPD]-6	—	9	—
S(3)-2	[PSB]-4	—	—	10	6
S(3)-3	[SPD]-3	—	—	10	—
S(4)-1	[SPD]-6	[PSB]-1	3,5	—	—
S(4)-2	[SPD]-1	—	—	—	—
S(4)-3	[SPD]-1	—	—	2	—
S(4)-4	[SPD]-5,6	—	—	12	—
S(4)-5	—	—	—	1, 6	—
S(4)-6	—	—	—	—	—
S(5)-1	—	—	—	—	—
S(5)-2	—	—	6	—	—
S(6)-1	—	—	[RFC]	—	2, 3
S(6)-2	—	—	[RFC]	—	—

*[SBD]: SECURE BY DEFAULT PRACTICES
[SPD]: SECURE PRODUCT DEVELOPMENT PRACTICES
[PSB]: PRO-SECURITY BUSINESS PRACTICES
[RFC]: RECOMMENDATIONS FOR CUSTOMERS

(6) Correspondence relationship with ANNEX I/II, EU CRA

Requirements	ANNEX I Part I	ANNEX I Part II	ANNEX II
S(1)-1	(1), (2)-a/g/h/i/j/k	(3)	—
S(1)-2	(1)	—	—
S(1)-3	(1)	—	—
S(1)-4	(1), (2)-d/e/f	—	—
S(2)-1	—	(6)	—
S(2)-2	(2)-l	(1)	—
S(2)-3	—	—	—
S(2)-4	(2)-b/f/m	(7)	4, 5, 7, 8-a/b/d
S(3)-1	(2)-a/j	(1), (2), (3), (6)	2
S(3)-2	(2)-a/c/j/l	(2), (4), (5), (7), (8)	8-c
S(3)-3	(2)-a/j/k	(2)	—
S(4)-1	—	—	—
S(4)-2	(2)-a/d/e/f/g/h/i	—	—
S(4)-3	(2)-a/g/h/i	—	—
S(4)-4	—	—	—
S(4)-5	—	—	—
S(4)-6	—	—	—
S(5)-1	—	—	—
S(5)-2	—	—	—
S(6)-1	—	—	—
S(6)-2	—	—	—

(7) Correspondence relationship with other documents

In the examples of additional measures described for the respective requirements, the following documents were referenced:

Requirements	Other related documents
S(1)-1	BSA
S(1)-2	BSA, CISA-D
S(1)-3	SP800-161, CISA-D, ISO15408
S(1)-4	ISMS, DSP
S(2)-1	SP800-161, CISA-D, BSA
S(2)-2	BSA, CISA-D, NSA
S(2)-3	BSA, SP800-161, Ministry of Internal Affairs and Communications, CISA-D, ISO15408, DSP, ISMS
S(2)-4	BSA, ISO15408, CISA-D
S(3)-1	SP800-161, BSA, ISMS, DSP, CISA-D
S(3)-2	CISA-D, BSA
S(3)-3	SP800-161, ISMS
S(4)-1	SP800-161, ISMS, CISA-D, BSA
S(4)-2	SP800-161, CISA-D, DSP, Japan-US-Australia-India Cybersecurity Partnership
S(4)-3	DSP, Japan-US-Australia-India Cybersecurity Partnership
S(4)-4	SP800-161, CISA-D, Japan-US-Australia-India Cybersecurity Partnership
S(4)-5	SP800-161, CISA-D, Japan-US-Australia-India Cybersecurity Partnership
S(4)-6	SP800-161, Japan-US-Australia-India Cybersecurity Partnership
S(5)-1	NSA, DSP, ISMS
S(5)-2	DSP, Japan-US-Australia-India Cybersecurity Partnership
S(6)-1	CISA-D
S(6)-2	BSA, CISA-D

5.8. Terminology

Agile development	Development process for updating software in stages by allocating SDLC phases into multiple development cycles and rapidly repeating the respective phases.
Build pipeline	Means of dividing the build process into multiple testing processes, and running phased execution. CI: One of practices of continuous integration (see DevSecOps).
Computer security incident response team (CSIRT)	Organization that responds to incidents.
Common Vulnerability Scoring System (CVSS)	Open and general-purpose evaluation technique for vulnerabilities in an information system. CVSS allows for quantitative comparison of the severity of vulnerabilities under specific conditions.
Hardening	Means of strengthening security by reducing system vulnerabilities and unnecessary functions.
Infrastructure as a service (IaaS)	Means of providing infrastructures such as networks and storage systems required to run information systems as services via the Internet.
Information and communication technology (ICT)	Generic term for information and communication technologies.
Integrated development environment (IDE)	Software into which functions required to develop software codes efficiently are integrated.
Indicator of compromise (IoC)	Traces and indicators of infringement such as a cyberattack.
Internet of Things (IoT)	Framework for connecting "things" such as sensor devices to the Internet.
Information Sharing and Analysis Center (ISAC)	This organization, started when respective private sector industries that make up critical infrastructures were encouraged to establish it to protect national critical information networks in the US, strives to promote information sharing on security, etc. by industry. In Japan, the Software ISAC, Finance ISAC, Transportation ISAC, etc. have been established.
Key performance indicator (KPI)	Quantitative indicator used to observe degrees of achievement of organizational goals.
Key risk indicator (KRI)	Indicator used to observe risk levels in an organization.
Malware Information Sharing Platform (MISP)	Open-source threat sharing platform aimed at accumulating and sharing IoCs, which are traces of cyberattacks such as IP addresses of destinations with which malware communicates. https://www.misp-project.org/
Open-source software (OSS)	Software whose source code is disclosed and allowed to be modified and changed.
Operational technology (OT)	Generic term for technologies that control and operate physical systems and facilities such as factories, plants, and buildings.

Platform as a service (PaaS)	Provision of platform functions designed for applications necessary to operate information systems as a service via the Internet.
Peer review Lead review	Activity in which developers and leaders of the same level diagnose and evaluate deliverables, making full use of their experience and know-how.
Product security incident response team (PSIRT)	Organization that strives to improve the security of products and services developed in-house and respond to incidents.
Regression testing	Test performed to check, after a program is changed, whether the program has problems in lines not changed.
Resilience	Term that can be translated as "elasticity," "resilience," "restorability," or "durability." The ability to limit damage and recover from an attack by taking appropriate countermeasures in the world of cybersecurity.
Risk modeling	Analytical technique for understanding the likelihood of possible threats, dangers, events, etc. that may occur, and identifying undesirable outcomes or problems. Software risk modeling employs threat modeling (a technique to study security measures from the perspective of protecting information assets through analysis in which characteristics of software, potential attackers, and attack methods are assumed), and in its process, uses an attack model (a model of possible attacker actions based on the types of attackers, attack surfaces, and attack methods).
Software as a service (SaaS)	Provision of information systems as services via the Internet.
Software Bill Of Materials (SBOM)	Technique for listing series of related elements such as components that make up software, their dependencies, and license data and managing them.
Software development life cycle (SDLC)	Development process that enables production of high-quality, low-cost software in a short period. Types available include waterfall development and agile development.
Secure by default	Philosophy or policy that makes software security functions and settings built in by default. For example, at the first stage where a product is purchased and used, creating a function to make access to other functions unavailable unless a sufficiently strong password is set that general users do not usually need unavailable by default is a concrete example that follows the philosophy of secure by default.
Secure by design	Philosophy or policy to assure information security from the software design stage. It may be referred to as security by design, but the terms are synonymous. The term "secure by design" encompasses "secure by default." In the "Secure by Design Software Principles and Approaches" published by the CISA in collaboration

	with international partners including National center of Incident readiness and Strategy for Cybersecurity (NISC,current National Cybersecurity Office (NCO)) to strive proactively for customer security assurance through the principle of secure by design, the following three software product security principles are advocated: Principle 1: Take Ownership of Customer Security Outcomes Principle 2: Embrace Radical Transparency and Accountability Principle 3: Lead From the Top Note that a similar term, "shift left," refers to incorporating security measures upstream in software development.
Security requirement	Specific requirement for security goals to be met at the time of development and implementation of a product or system.
Service level agreement (SLA)	Content agreed between service provider and service user regarding the scope, content, and goals to be attained of the service.
Security operation center (SOC)	Specialized organization that monitors network devices, server logs, etc. to detect and analyze cyberattacks and their precursors.
Software supply chain	Interdependency between software life cycle related to all of software design, development, supply and operation, related organizations, and software.
Toolchain	Set of software tools that have functions required for software development. Aimed at improving the efficiency of development work by linking respective tools.
Value stream mapping (VSM)	Lean manufacturing method to analyze, design, and manage sequences of materials, information, etc. required in development and operation processes for delivering products such as software.
Walk-through	Desk review conducted by bringing those who are concerned with development together as well as creators of deliverables to improve the quality of deliverables such as specifications. Method to find problems in specifications of a system and programs in a system.
DevSecOps	Coined word combining the initials of Development, Security, and Operations, or the practice of integrating security tests in all phases of software development processes. The "CI/CD pipeline" implements part of this concept, and is a mechanism that continuously updates software in phases. The mechanism is automatically deployed through verification with automatic building and testing. Note that CI/CD stands for Continuous Integration / Continuous Delivery or Deployment.

6. Organizational system for examining the Guidelines

Study Group on the Roles Required of Cyber Infrastructure Providers

The "Study Group on the Roles Required of Cyber Infrastructure Providers" was formed in September 2024 as a joint working group of the Cross-Sectoral Sub-working Group and Critical Infrastructure Expert Examination Committee, Cybersecurity Strategy Headquarters, Study Group on Industrial Cybersecurity WG1, Ministry of Economy, Trade and Industry. The group has held discussions on the wide range of roles expected of cyber infrastructure providers to improve the resilience of software supply chains. (Following the abolition of the Critical Infrastructure Expert Examination Committee in July 2025, it was repositioned as a joint working group of the Cross-Sectoral Sub-working Group and Critical Infrastructure Expert Examination Committee, Cybersecurity Strategy Headquarters, Study Group on Industrial Cybersecurity WG1, Ministry of Economy, Trade, and Industry, and National Cybersecurity Office.)

We have formulated the Guidelines through discussions in the study group; these guidelines outline the responsibilities of cyber infrastructure providers and customers regarding the software design, development, supply, and operation, along with the requirements for fulfilling their responsibilities (specific measures by role) and methods to disseminate the Guidelines (such as a structural implementation for self-declaration of conformity).

<List of members>

*Titles omitted, as of March 4, 2026

	ABE Kyoichi	Senior Advisor, LEON TECHNOLOGY, INC.
	INAGAKI Ryuichi	Bengoshi (Attorney at law), Inagaki Ryuichi Law Firm
	KAMODA Hiroaki	Head of Security and Network Division, Solution Sector, NTT DATA Japan
	KITANI Hiroshi	Chairman of Cyber Security Subcommittee, Japan Information Technology Services Industry Association (JISA); Assessment Department, Advanced Security Technical Division, Cyber Security Technology Development Group, Canon IT Solutions Inc.
	TATEISHI Toshiaki	Board member, Information Technology Federation of Japan; Vice Chairman, Japan Internet Providers Association
	TSUDA Hiroshi	Fellow SVP, Fujitsu Research, Fujitsu Limited
Chair	DOI Norihisa	Professor Emeritus, Keio University
	BANDO Naoki	Fellow, Software Association of Japan (SAJ); Co-Representative, Software ISAC
	HIDAKA Shoji	Executive Officer, Japan Cloud Industry Association (ASPIC)
	FUCHIGAMI Shinichi	Corporate Executive CISO and General Manager, Cybersecurity Strategy Department, NEC Corporation
	FURUTA Tomoji	Project General Manager, Information Security and Trust

	Management Division, TOYOTA MOTOR CORPORATION
YAMAGUCHI Masafumi	Business Strategy Promotion Division Manager, NRI Secure Technologies Ltd.

(Secretariat)
Ministry of Economy, Trade and Industry, National Cybersecurity Office (NCO)

(Observers)
National Police Agency, Ministry of Internal Affairs and Communications, Ministry of Health, Labour and Welfare, Defense Equipment Agency, Digital Agency, Japan Federation of Medical Devices Associations

<Summary>

Date	Agenda/Summary
First meeting (September 24, 2024)	[Agenda] Discussion on responsibilities and requirements, and how to proceed with examination [Summary of the meeting] Discussion on the "responsibilities" and "requirements" expected of cyber infrastructure providers, as well as how to proceed with the examination of Guidelines (draft)
Second meeting (December 17, 2024)	[Agenda] Deliberation of the Guidelines (draft) based on the results of literature surveys and hearings, and discussions regarding the Guidelines Annex policies [Summary of the meeting] Discussion on the Guidelines (draft) and dissemination measures for the Guidelines
Third meeting (February 18, 2025)	[Agenda] Discussion on approval of the Guidelines (draft) and examination of dissemination policies in the future [Summary of the meeting] Deliberation on updating of the Guidelines (draft) and discussion on efforts and dissemination measures to promote implementation of the Guidelines (draft).
Fourth meeting (February 5, 2026)	[Agenda] Deliberation on the Guidelines (revised draft version) and the Requirements Checklist based on the results of pilot tests and hearings, and discussions on dissemination policy [Summary of the meeting] Deliberation on revision details of the Guidelines (draft) and the Requirements Checklist, and discussions on efforts and dissemination measures to promote implementation of the Guidelines (draft)

Fifth meeting (February 26 to March 4, 2026)	[Agenda] Discussions on the deliberations concerning the Guidelines (draft) and the Requirements Checklist [Summary of the meeting] Deliberations on the Guidelines (draft) and the Requirements Checklist

Requirements checklist

	Requirement		Requirements for fulfilling responsibilities						
Check!	Minimum	Standard	Requirement ID	Itemized requirements	Individual requirements	Developer	Supplier	Operator	Customer
	✓	✓	S(1)-1.1	Risk-based security requirements definition	Perform risk-based analysis and assessment of the software to be developed or the system/service using the software, and define security requirements that serve as mitigation measures.	✓			
	✓	✓	S(1)-1.2	Design review	Through a review of the software design, confirm that it meets all security requirements and adequately addresses identified risks, and apply the review results.	✓			
		✓	S(1)-1.3	Risk response records	Maintain records of design decisions, responses to risks, and approved exceptional measures for audit and maintenance purposes throughout the software life cycle.	✓			
		✓	S(1)-1.4	Periodic risk-based review	Review all approved exceptions to security requirements and software design, as well as the results of the risk-based analysis and assessment created during the software design, and periodically check whether risks are being addressed appropriately.	✓			
	✓	✓	S(1)-2.1	Definition of secure development process	Define processes related to secure coding, secure build, and secure by default by considering secure coding perspectives, the build timing and method, the use of automation tools, and training.	✓			
	✓	✓	S(1)-2.2	Secure build	Generate and build code using a compiler, an interpreter, and build tools that provide functions to improve the security of executable formats.	✓			
	✓	✓	S(1)-2.3	Verification and feedback	Identify root causes of problems discovered through verification by review and analysis, and then feed the results back to the processes.	✓			
	✓	✓	S(1)-2.4	Codebases	For objects subject to review and analysis, not only source codes but also codes in various formats (such as configuration files) that the organization determines to be readable should be targets.	✓			
	✓	✓	S(1)-3.1	Test planning	Based on threat models and risk analysis, determine a test scope and test method, and develop a test plan.	✓			
	✓	✓	S(1)-3.2	Test method	Include functional testing, vulnerability testing, fuzzing, penetration testing, etc. in the test method.	✓			
	✓	✓	S(1)-3.3	Test implementation	Design and implement tests according to the test plan, and document the test results.	✓			
	✓	✓	S(1)-3.4	Response to problems	Incorporate all problems identified through testing and recommended countermeasures into the development team's workflows to solve them.	✓			
	✓	✓	S(1)-4.1	Asset management	Operators arrange asset management procedures and asset lists related to assets handled by systems and services as well as assets that constitute the systems and services.			✓	
		✓	S(1)-4.2	Development of a monitoring environment	Operators separate systems appropriately to minimize the potential impact of a risk when it occurs, and arrange a monitoring environment to monitor risks that are important to protect assets by means of software.			✓	
		✓	S(1)-4.3	Arrangement of a security mechanism	An appropriate security mechanism is arranged that allows software and systems and services to which the software is applied to protect and monitor the confidentiality and integrity of information assets and data in operating environments or resources such as digital infrastructure.	✓		✓	
	✓	✓	S(1)-4.4	Monitoring and evaluation	Operators monitor the operation of mechanisms applied to software that provides important services, periodically conduct security assessments, and integrate them into the risk management framework of the organization.			✓	
	✓	✓	S(2)-1.1	Arrangement of software components	With respect to commercial, open-source, and other third-party software components procured from outside, adopt those that are highly secure and meet the defined in-house requirements.	✓			
	✓	✓	S(2)-1.2	Development and maintenance of software components	When the software components are not procured from outside, develop highly secure software components in-house in accordance with established in-house security standards and practices, and maintain them.	✓			
	✓	✓	S(2)-1.3	Risk assessment of software components	Acquire and analyze information regarding locations from where the respective software components are obtained and assess the risks resulting from the components.	✓			
	✓	✓	S(2)-1.4	Confirmation of publicly known vulnerabilities of software components	Regularly check for publicly known vulnerabilities and periods during which respective software components are supported.	✓			
	✓	✓	S(2)-1.5	Update of software components	Implement a process to update the respective software components to the new version securely.	✓			
	✓	✓	S(2)-2.1	Protection of codebases	To protect codebases in all forms from unauthorized access and tampering, store the codes and configuration information in a repository and implement access control based on the principle of least privilege so that only authorized personnel, tools, and services can access it.	✓	✓		
	✓	✓	S(2)-2.2	Archiving of releases	Archive the respective software releases to protect them so that vulnerabilities identified following release can be analyzed and identified.	✓	✓		
	✓	✓	S(2)-2.3	Sharing of release provenance data	Collect, protect, maintain, and share provenance data for all components of the respective software releases.	✓	✓		
	✓	✓	S(2)-3.1	Agreement on security requirements	Include explicit security requirements in contracts or policies to be shared with third parties that provide software (including commercial software components for use in in-house software) or services.	✓	✓	✓	
		✓	S(2)-3.2	Response to supply chain security requirements	Respond to supply chain security requirements equivalent to those adopted by the organization that receives or acquires third-party software or services that it provides.	✓	✓		
		✓	S(2)-3.3	Establishment of a response process to risks that do not meet security requirements	Arrange a process to respond to risks in the case in which there are security requirements that third-party software or services to be received or acquired do not meet.	✓	✓	✓	
	✓	✓	S(2)-4.1	Secure introduction, configuration, operation, modification, disposal, and termination	Ensure that software users can continuously use information for securely introducing, configuring, and operating software, as well as information related to the impact of changes, disposal, termination of provision, and termination of use.	✓	✓		
	✓	✓	S(2)-4.2	Provision of integrity verification information	Ensure that software users can continuously use information that is necessary for verifying the integrity and completeness of the software.	✓	✓		
	✓	✓	S(3)-1.1	Establishment of a vulnerability response system	Establish a policy for the disclosure and remediation of vulnerabilities of software products, establish a system for responses to vulnerabilities (including responses to incidents) to support the policy, and define necessary roles, responsibilities, and processes.	✓		✓	
	✓	✓	S(3)-1.2	Communication plan	Establish a communication plan for all stakeholders.	✓		✓	
	✓	✓	S(3)-1.3	Vulnerability information collection	Collect new information regarding vulnerabilities through searches of public information, notifications from software users, the acquisition of external threat information, reviews of system configuration data, and other methods.	✓		✓	
	✓	✓	S(3)-1.4	Identification of undetected vulnerabilities	Conduct software code review, analysis, and testing on an ongoing or regular basis to identify undetected vulnerabilities (including improper settings) to be solved.	✓		✓	
	✓	✓	S(3)-2.1	Vulnerability analysis	Developers collect information necessary to understand the risks associated with the impact of each remaining vulnerability and analyze each vulnerability to plan remediations or other responses to risks.	✓			
	✓	✓	S(3)-2.2	Risk response to vulnerabilities	Developers create a plan for risk responses for each vulnerability and implement it.	✓			
	✓	✓	S(3)-2.3	Security recommendations	Developers prepare security recommendations, provide the information to the supplier of the released software, and create a report as specified by the relevant systems. In addition, operators implement deployment in accordance with security recommendations.	✓	✓	✓	
		✓	S(3)-3.1	Identification of root causes	Analyze an identified vulnerability to determine its root causes and proactively take countermeasures.	✓		✓	
		✓	S(3)-3.2	Process improvement	Review development and operation processes for the entire software life cycle and revise them as necessary to prevent root causes from recurring or reduce the possibility of their recurrence through software updates or new software creation.	✓		✓	
		✓	S(4)-1.1	Definition of roles and responsibilities	Define roles and responsibilities covering the entire software development life cycle.	✓	✓	✓	
	✓	✓	S(4)-1.2	Management's commitment	Make management's commitment to secure development known to all personnel, and educate them on the importance of secure development and operation to the organization.	✓	✓	✓	
		✓	S(4)-1.3	Agreement on roles and responsibilities	Confirm that all personnel are aware of and agree to their roles and responsibilities.	✓	✓	✓	
		✓	S(4)-1.4	Training for each role	Create a training plan for each role and implement it so that all personnel can be trained according to their level of proficiency and role.	✓	✓	✓	
		✓	S(4)-1.5	Review of roles and training	Review roles and training regularly.	✓	✓	✓	

Requirements checklist

	Requirement		Requirements for fulfilling responsibilities						
Check!	Minimum	Standard	Requirement ID	Itemized requirements	Individual requirements	Developer	Supplier	Operator	Customer
	✓	✓	S(4)-2.1	Definition of a software development policy	Identify all security requirements for software development infrastructures and processes (including requirements related to EOL), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.	✓			
	✓	✓	S(4)-2.2	Definition and maintenance of a software security policy	Define a policy that specifies all security requirements that must be met by the software developed by an organization, and maintain the requirements throughout the SDLC.	✓			
	✓	✓	S(4)-2.3	Sharing of cost recognition and budgeting	Secure necessary budgets to ensure security based on a policy.	✓			
		✓	S(4)-3.1	Definition of a software service operation policy	Identify all security requirements for service operation infrastructures and processes to which the software is applied (including requirements related to EOS and disposal), and define a security policy for maintenance throughout the SDLC in compliance with laws and regulations.			✓	
		✓	S(4)-3.2	Definition and maintenance of a service security policy	Define a policy that specifies all security requirements that services to which the software is applied must meet, and maintain the requirements throughout the SDLC.			✓	
		✓	S(4)-3.3	Audit based on an operational policy	Confirm through an audit that the protection of service operation infrastructures and processes and security requirements for service are maintained throughout the SDLC in accordance with policy-based governance.			✓	
	✓	✓	S(4)-4.1	Definition and tracking of security verification criteria	Define software security verification criteria and track the entire SDLC.	✓		✓	
	✓	✓	S(4)-4.2	Support for decision-making based on security verification criteria	Implement processes and mechanisms for collecting and protecting information necessary to support decision-making based on security verification criteria.	✓		✓	
		✓	S(4)-4.3	Audit based on security verification criteria	Track the entire SDLC and verify through audits that the intended effects are achieved with governance to ensure conformance to security verification criteria.	✓		✓	
	✓	✓	S(4)-5.1	Designation of tools and toolchains	Identify tools that are effective in mitigating specific risks and determine means of integrating toolchain components mutually.	✓			
	✓	✓	S(4)-5.2	Deployment, operation, and maintenance of tools and toolchains	Deploy, operate, and maintain tools and toolchains in accordance with security practices.	✓			
	✓	✓	S(4)-5.3	Tool configuration and evidence generation	Configure tools to generate evidence regarding support for secure software development practices defined in-house.	✓			
	✓	✓	S(4)-6.1	Isolation and protection of environments	Isolate and protect the respective environments related to software development.	✓			
	✓	✓	S(4)-6.2	Protection of development endpoints	Protect and strengthen endpoints designed for respective developers to perform development-related tasks using a risk-based approach.	✓			
		✓	S(5)-1.1	Establishment of an organizational system for information sharing	Establish an organizational structure for information sharing between private companies, relevant authorities, and specialized organizations to improve the security of software products and services.	✓	✓	✓	
	✓	✓	S(5)-1.2	Provision of important security-related information	Select and identify essential and important security-related information that is specific to the industry and provide it to partners in the supply chain.	✓	✓	✓	
	✓	✓	S(5)-1.3	Use of vulnerability information notification services	Use vulnerability information notification services to share vulnerability information efficiently.	✓	✓	✓	
		✓	S(5)-2.1	Utilization of cooperation systems	To improve the security of software products and services, make use of communities and cooperation systems aimed at improving software security, in which external businesses, customers, and specialized organizations participate.	✓	✓	✓	
		✓	S(5)-2.2	Contribution to cooperation systems	When participating in a community or cooperation system, actively participate in activities to contribute to the cooperation system.	✓	✓	✓	
	✓	✓	S(6)-1.1	Risk management	Implement risk management in which the customer's independent and proactive efforts are integrated with efforts based on a contract with cyber infrastructure providers.				✓
	✓	✓	S(6)-1.2	Resource arrangement	Allocate and develop resources to respond proactively to known vulnerabilities and implement mitigation measures (including SBOM utilization).				✓
		✓	S(6)-1.3	Utilization of cooperation systems	Proactively participate in and utilize communities and collaborative systems aimed at improving software security.				✓
	✓	✓	S(6)-2.1	Definition of security requirements	Define security requirements for incorporating security functions into software design plans and present them to cyber infrastructure providers before procuring and deploying software.				✓
	✓	✓	S(6)-2.2	Disclosure of security practice requirements	Disclose security practice requirements for cyber infrastructure providers before procuring and deploying software.				✓
	✓	✓	S(6)-2.3	Decision-making based on risk assessment	When procuring and introducing software, make decisions based on risk assessment.				✓
	✓	✓	S(6)-2.4	Budget securement	Continuously secure budgets related to introduction, operation, migration, disposal, risk response, and related contracts, considering software life cycles.				✓

Requirements checklist (role/phase)

	Requirements		Requirements for fulfilling responsibilities						Life cycle phase														
Check!	Minimum	Standard	Requirement ID	Itemized requirements	Developer	Supplier	Operator	Customer	Requirement definition	Design	Development	Build	Testing	Release	Distribution	Deployment	Operation	Monitoring	Feedback	Analysis	Planning	Arrangement of human resources, processes, and technologies	Strengthening of relationships between cyber infrastructure providers and stakeholders
	✓	✓	S(1)-1.1	Risk-based security requirements definition	✓				✓												✓		
	✓	✓	S(1)-1.2	Design review	✓					✓													
		✓	S(1)-1.3	Risk response records	✓					✓	✓	✓	✓	✓						✓	✓		
		✓	S(1)-1.4	Periodic risk-based review	✓															✓	✓		
	✓	✓	S(1)-2.1	Definition of secure development process	✓						✓	✓											
	✓	✓	S(1)-2.2	Secure build	✓							✓											
	✓	✓	S(1)-2.3	Verification and feedback	✓						✓	✓											
	✓	✓	S(1)-2.4	Codebases	✓						✓	✓											
	✓	✓	S(1)-3.1	Test planning	✓								✓								✓		
	✓	✓	S(1)-3.2	Test method	✓								✓										
	✓	✓	S(1)-3.3	Test implementation	✓								✓										
	✓	✓	S(1)-3.4	Responses to problems	✓								✓										
	✓	✓	S(1)-4.1	Asset management			✓										✓						
		✓	S(1)-4.2	Development of a monitoring environment			✓										✓	✓					
		✓	S(1)-4.3	Arrangement of a security mechanism	✓		✓										✓	✓					
	✓	✓	S(1)-4.4	Monitoring and evaluation			✓											✓	✓	✓			
	✓	✓	S(2)-1.1	Arrangement of software components	✓				✓	✓	✓												
	✓	✓	S(2)-1.2	Development and maintenance of software components	✓						✓												
	✓	✓	S(2)-1.3	Risk assessment of software components	✓					✓										✓			
	✓	✓	S(2)-1.4	Confirmation of publicly known vulnerabilities of software components	✓															✓	✓		
	✓	✓	S(2)-1.5	Updating of software components	✓						✓												
	✓	✓	S(2)-2.1	Protection of codebases	✓	✓								✓									
	✓	✓	S(2)-2.2	Archiving of releases	✓	✓								✓	✓					✓			
	✓	✓	S(2)-2.3	Sharing of release provenance data	✓	✓								✓	✓								
	✓	✓	S(2)-3.1	Agreement on security requirements	✓	✓	✓		✓														
		✓	S(2)-3.2	Responses to supply chain security requirements	✓	✓			✓	✓													
		✓	S(2)-3.3	Establishment of a response process for risks that do not meet security requirements	✓	✓	✓		✓	✓													
	✓	✓	S(2)-4.1	Secure introduction, configuration, operation, modification, disposal, and termination	✓	✓								✓	✓	✓							
	✓	✓	S(2)-4.2	Provision of integrity verification information	✓	✓								✓	✓	✓							
	✓	✓	S(3)-1.1	Establishment of a vulnerability response system	✓		✓														✓		
	✓	✓	S(3)-1.2	Communication plan	✓		✓														✓		
	✓	✓	S(3)-1.3	Vulnerability information collection	✓		✓											✓		✓			
	✓	✓	S(3)-1.4	Identification of undetected vulnerabilities	✓		✓			✓			✓							✓			
	✓	✓	S(3)-2.1	Vulnerability analysis	✓														✓	✓			
	✓	✓	S(3)-2.2	Risk responses to vulnerabilities	✓					✓	✓	✓	✓							✓	✓		
	✓	✓	S(3)-2.3	Security recommendations	✓	✓	✓							✓	✓		✓						
		✓	S(3)-3.1	Identification of root causes	✓		✓			✓	✓									✓			
		✓	S(3)-3.2	Process improvement	✓		✓													✓	✓		
		✓	S(4)-1.1	Definition of roles and responsibilities	✓	✓	✓															✓	
	✓	✓	S(4)-1.2	Management's commitment	✓	✓	✓															✓	
		✓	S(4)-1.3	Agreement on roles and responsibilities	✓	✓	✓															✓	
		✓	S(4)-1.4	Training for each role	✓	✓	✓															✓	
		✓	S(4)-1.5	Review of roles and training	✓	✓	✓															✓	
	✓	✓	S(4)-2.1	Definition of a software development policy	✓																	✓	
	✓	✓	S(4)-2.2	Definition and maintenance of a software security policy	✓																	✓	
	✓	✓	S(4)-2.3	Sharing of cost recognition and budgeting	✓																	✓	
		✓	S(4)-3.1	Definition of a software service operation policy			✓															✓	
		✓	S(4)-3.2	Definition and maintenance of service and security policies			✓															✓	
		✓	S(4)-3.3	Audit based on an operational policy			✓											✓				✓	
	✓	✓	S(4)-4.1	Definition and tracking of security verification criteria	✓		✓															✓	
	✓	✓	S(4)-4.2	Support for decision-making based on security verification criteria	✓		✓															✓	
		✓	S(4)-4.3	Audit based on security verification criteria	✓		✓											✓				✓	
	✓	✓	S(4)-5.1	Designation of tools and toolchains	✓																	✓	
	✓	✓	S(4)-5.2	Deployment, operation, and maintenance of tools and toolchains	✓																	✓	
	✓	✓	S(4)-5.3	Tool configuration and evidence generation	✓																	✓	
	✓	✓	S(4)-6.1	Isolation and protection of environments	✓																	✓	
	✓	✓	S(4)-6.2	Protection of development endpoints	✓																	✓	
		✓	S(5)-1.1	Establishment of an organizational system for information sharing	✓	✓	✓																✓
	✓	✓	S(5)-1.2	Provision of important security-related information	✓	✓	✓																✓
	✓	✓	S(5)-1.3	Use of vulnerability information notification services	✓	✓	✓																✓
		✓	S(5)-2.1	Utilization of cooperation systems	✓	✓	✓																✓
		✓	S(5)-2.2	Contribution to cooperation systems	✓	✓	✓																✓
	✓	✓	S(6)-1.1	Risk management				✓															
	✓	✓	S(6)-1.2	Resource arrangement				✓															
		✓	S(6)-1.3	Utilization of cooperation systems				✓															
	✓	✓	S(6)-2.1	Definition of security requirements				✓															
	✓	✓	S(6)-2.2	Disclosure of security practice requirements				✓															
	✓	✓	S(6)-2.3	Decision-making based on risk assessment				✓															
	✓	✓	S(6)-2.4	Budget securement				✓															