TECHNICAL SPECIFICATION

ISO/IEC TS 19249

First edition
2017-10

Information technology — Security techniques — Catalogue of architectural and design principles for secure products, systems and applications

Technologies de l'information — Techniques de sécurité — Catalogue des principes architecturaux et conceptuels pour la sécurisation des produits, systèmes et applications

Reference number
ISO/IEC TS 19249:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

Introduction

This document describes architectural and design principles that may be used in the development of secure products, systems and applications.

Each principle is described in a structured way, characterizing the principle itself, describing how it can be used, how it can support security, how it may help in the security assessment of the product, system or application, as well as its dependency on other principles described in this document.

Examples are provided for each principle on how it may be implemented, how it may contribute to security properties and functions and what other aspects have to be taken into account in the example provided to also address non-security related requirements like usability and performance.

A guideline is provided linking this to security evaluations performed using ISO/IEC 15408 (all parts) and ISO/IEC 18045 and addresses both developers and evaluators of secure products, systems and applications.

Information technology — Security techniques — Catalogue of architectural and design principles for secure products, systems and applications

1 Scope

This document provides a catalogue of architectural and design principles that can be used in the development of secure products, systems and applications together with guidance on how to use those principles effectively.

This document gives guidelines for the development of secure products, systems and applications including a more effective assessment with respect to the security properties they are supposed to implement.

This document does not establish any requirements for the evaluation or the assessment process or implementation.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15408-1, *Information technology — Security techniques — Evaluation criteria for IT security — Part 1: Introduction and general model*

ISO/IEC 15408-2, *Information technology — Security techniques — Evaluation criteria for IT security — Part 2: Security functional components*

ISO/IEC 15408-3, *Information technology — Security techniques — Evaluation criteria for IT security — Part 3: Security assurance components*

ISO/IEC 18045, *Information technology — Security techniques — Methodology for IT security evaluation*

3 Terms and definitions

For the purposes of this document, terms and definitions given in ISO/IEC 15408-1, ISO/IEC 15408-2, ISO/IEC 15408-3 and ISO/IEC 18045 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
security property
property of a system or application that is crucial to achieve the security objectives defined for the system or application

3.2
security relevant attribute
attribute assigned to an object, subject or user of a system or application that is used to enforce a security policy

3.3
privilege
specific security relevant attribute for a subject or user that allows that user or subject to perform dedicated activities bound to that security relevant attribute

3.4
transparent encryption service
service provided by the system that is either always on or that can be switched on by an administrator which automatically encrypts and decrypts data on persistent storage or on communication links and where the user that stores or transmits data is not aware that this encryption service is performed

EXAMPLE Automatic disk encryption services (like self-encrypting disks); tunnelling a communication link through an IPSec or TLS protected communication link.

4 Architectural principles for secure products, systems and applications

4.1 General

Building a secure product, system or application requires not only the implementation of functional requirements but also an architecture that allows for the effective enforcement of specific security properties the product, system or application is supposed to enforce. The ability to withstand attacks the product, system or application may face in its intended operational environment is highly dependent on an architecture that prohibits those attacks or – if they cannot be prohibited – allows for detection of such attacks and/or limitation of the damage such an attack can cause. Structuring the product, system or application into dedicated domains that are isolated from each other and can only communicate using well defined communication paths where specific security policies can be enforced is a generic architectural principle that can be applied. The architectural structure imposed by such domains has to be carefully selected to support the enforcement of security properties and requirements but also assist in the implementation of non-security related requirements. Such a structure should also not cause unacceptable degradation of the performance of the product, system or application. Finding the right balance in the architectural structure that supports all requirements for a product, system or application is the main topic the architectural principles defined in this clause can be used in co-operation with the related design principles.

There are a few fundamental principles for security that need to be observed regardless of any architecture or design principle. Those have been known for a long time and were first documented in a study performed on behalf of the US Government in 1972.[1] Those fundamental principles are:

— the implementation of security functions needs to be such that those functions cannot be bypassed or tampered with by untrusted code;

— the security functions need to be always invoked when required to enforce the security policy;

— the part that implements the security functionality need to be small enough to be analysed for correctness and potential security critical side effects.

In a later report [2], the parts of a product that need to be trusted for enforcing the security policy and that satisfy the security principles above was called the "Trusted Computing Base" or TCB.

A number of design principles discussed in this document were previously introduced in [3] in 1974. Since then, IT products and systems have significantly grown in complexity and are now often distributed, rely on external service providers, and cover a wider spectrum of security functionality. This requires more complex architectures and designs and this document provides a description of architectural and design principles with their security-related aspects.

4.2 Domain separation

4.2.1 General

A domain is a concept for encapsulating components, data and/or programs into separate entities which can be managed separately including assigning privileges and other security attributes to each domain. There may be defined communication channels between domains, implemented in a way that allows to control which domains can communicate with which other domain and also allow a domain receiving a request from another domain over a communication channel to identify from which domain the request comes, allowing to base its reaction also on the identity of the requesting domain.

There may be reasons for structuring a system, application etc. other than security. This document only takes security-related reasons for structuring into account.

From a security point of view, structuring a product, system or application into separated domains may be an effective method to keep functions and objects requiring a dedicated set of privileges, access rights together or to isolate critical functions and objects from less critical ones. Domain separation is especially important and helpful if specific security properties can be mapped to dedicated domains and the proof that a system or product satisfies those properties can be restricted to proofing this for those dedicated domains.

Domain separation is also often associated with a distributed design where the individual domains are implemented on physically separated parts of a system or an application and communicate via network connections. In such architectures a domain may provide some general service that is used by multiple systems. Examples of security services provided in such a way are directory services, centralized authentication services, certificate management services or centralized access control services.

4.2.2 Principles for defining domain structures

There are not only security reasons that should be taken into account when structuring an application or a system into separate domains. In this document only the security relevant reasons are listed.

The security reason for defining domains is to separate components of an application or system with common security relevant attributes like privileges, access to files, etc. from other components with different security relevant attributes. A good domain structure allows each domain to execute with the least privileges it requires to perform its task. In addition the isolation of domains with well-defined and controlled interfaces and interactions between them allows for better error detection, limited error propagation and implementation of a defence-in-depth strategy. Domains should be separated from each other such that the interference between the domains is limited and can be controlled. The method used for isolation depends on the operational environment. Individual domains may be implemented on different physical systems interconnected by some network; they may be implemented by different virtual machines that communicate via mechanisms provided by the underlying virtual machine monitor; or they may be implemented as separate processes operating on an operating system. In those cases the separation mechanism is provided by the underlying hardware, virtual machine monitor or operating system. In other cases the application or system itself may implement a layer that provides the general separation mechanism itself (often using supporting features of the underlying hardware or software).

It is often a good strategy to define domains that are exposed to an attack surface with just the minimum set of privileges they need for their operation. Trust in such domains by other domains should also be limited which requires other domains to perform additional checks on data they receive from such a domain with a lower level of trust.

4.2.3 Principles for defining inter-domain communication

Inter-domain communication is required for the individual domains to be able to cooperate. Since domains have different privileges and may be exposed to different attack surfaces, the communication

between different domains needs to be controlled by a policy that takes the threats associated with the differences in privileges and attack surfaces into account. Such a policy therefore needs to identify:

— what one domain can trust another domain for;

— what one domain can assume about data received from another domain (e.g., if data attributes can be trusted, if data has been parsed for its syntax and structure, if data has been kept confidential);

— which security validations need to be performed;

— when to accept or reject data received.

4.2.4 Security policies that may be enforced using domain separation

Domain separation can be used as a basis to define information flow policies or workflow policies by defining the way domain can interact with each other. Policies may be defined based on the identity of domains or their attributes like their privileges. Also the level of trust into the individual domains may be used for policy enforcement.

4.2.5 Examples

4.2.5.1 Structuring a network into domains

Networks may be structured into domains (segments) where the communication between different domains is regulated by gateways like routers with firewall functions or more complex application layer filtering appliances. Routing as well as techniques like VPN or VLAN may be used for defining the communication paths between domains while firewalls and gateways perform protocol and content related controls. Information flow control policies based on data labels can also be effectively enforced by domain separation where sample "trusted gateways" control the interfaces between different "single label" domains and ensure that data flows to a domain only when the label corresponds to the label of the domain.

4.2.5.2 Structuring an application into domains

Similar to networks, single applications can be structured into domains, which in this case are different processes. Those processes may either be a single physical system or distributed on multiple systems within a network. On a single system the processes are usually separated by the operating system or virtual machine monitor, which also provide the capability for defining and using communication paths between those processes.

4.2.5.3 Structuring data into domains

Data can also be structured into domains based on the importance of the data and its protection requirements. Critical data can then be put into a domain with a high level of protection like one that can only be accessed by processes with high privileges or one that is kept redundant to avoid data loss. Another method for structuring data into different domains is to encrypt data in each domain using a different encryption key. Whoever wants to get access to the information is required to have access to the key of the domain for decryption. Domains for data may be implemented by different files (which allow for different access control attributes), implemented by different disks with different reliability/integrity properties (like RAID levels) or make use of a DBMS as data storage that provides additional security like transaction safety.

4.2.6 Considerations for evaluation

In an evaluation, the aspects that need to be considered first are the mechanisms used to separate the domains and the mechanisms used for inter-domain communication.

Mechanisms that enforce domain separation, strength of domain separation and management of domains (if applicable) like creation, deletion of domains or changing the privileges of domains are the focus when evaluating the security aspects of domain separation. The aspects to be considered are:

— the strength of the separation mechanisms that separate domains such that the only way domains can influence each other is via the defined communication mechanisms;

— whether the domains and their privileges are static or dynamic (if they are dynamic, a more complex model needs to be established and evaluated showing that the security properties of the system cannot be violated by the dynamics allowed).

4.3 Layering

4.3.1 General

Layering is a hierarchical architecture where functions of one layer use functions of the next lower layer and provide functions that are used by the next higher layer in the hierarchy. Usually in a layered architecture the lowest layer provides very basic simple functions that are then used by the next higher layer to implement more complex functions, then used by the next layer to provide even more complex functions, and so on. Each layer abstracts from the functions provided by lower layers allowing to use those functions of lower layers only by the functions it provides. This allows each layer to also implement its own security policy – provided the layering architecture is implemented in a way that does not allow a layer to bypass the functions of the next lower layer and use functions provided two more layers down directly.

Layering is widely used in the design of network protocols and also in the design of applications. It is not primarily used for security reasons, but for supporting maintainability and extensibility – aspects that also support security and assurance.

4.3.2 Principles for defining layers

From a security point of view, layers provide a mechanism allowing the distribution of security functionality throughout the different layers depending on the abstraction of the functions and objects provided by each layer as well as the protection of functions implemented in hierarchically lower layers from being tampered with or bypassed by functions implemented in hierarchically higher layers.

In order to protect hierarchically lower layers, each layer (or a set of adjacent layers) may be implemented as its own domain with hierarchically lower layers having more privileges than higher layers.

4.3.3 Principles for Interfaces exposed by a layer

A layer usually implements higher level services by using functions of the next lower layer. This implies that all the abstractions required for a layer to implement its functionality can be built using interfaces provided by the next lower layer. In some cases a layer may be allowed to access functions of not just the next lower layer but also functions of layers lower than just the adjacent one.

4.3.4 Security policies that may be enforced using layering

Layering is not primarily an architectural principle associated with specific security policies, but it may be used efficiently for implementing security functionality within a layer that is transparent for higher layers. Transparent encryption services are one example of such a security service that may be implemented in a layer. This is quite common, e.g., for file systems as well as for cryptographic protocols. Also other security services that support integrity, access control or redundancy providing services can be implemented in a dedicated layer in a transparent way.

4.3.5 Examples

4.3.5.1 File system layering

For example in a layered implementation of a file system, the lowest layer may just provide access to raw sectors of a disk and its only security functionality may be to ensure the integrity of the disk sectors. A second layer may then implement virtual disks based on the raw disk sector layer and provide additional security functionality like a basic access control to those virtual disks (e.g., by allowing it to assign them to partitions or applications) or even implement full disk encryption functionality for those virtual disks or by implementing enhanced integrity mechanisms.

A third layer may then implement a file system on those virtual disks where the file system provides sophisticated access control functionality (e.g., based on access control list and/or other file security attributes) and potentially file based encryption.

A fourth layer on top of such a file system may be implemented by an application where the data of files is presented to individual users of the application implementing its own access control or other security functions based on individual records in the files.

4.3.5.2 Cryptographic functions

Also cryptographic functions can be implemented in a layered way. As an example the following layers may be defined.

— The lowest layers implements the basic cryptographic primitives like the encryption algorithms, hash algorithms, and random number generation.

— The next higher layer uses those functions to implement digital signature generation, digital signature verification, message authentication codes, key generation, key exchange, and key management etc.

— The next layer implements functions like certificate validation, certificate generation, and key wrapping.

4.3.6 Considerations for evaluation

Layering may be used within a single trust domain to structure functions within this domain or it may implement layers as their own domains. In the first case, no specific evaluation aspects need to be considered. In the case where layers are implemented as separate domains, the evaluation needs to verify the domain separation aspects are satisfied and in addition that layers are built correctly on each other. While it is not forbidden that a layer uses functions not just from the next lower layer but also from deeper layers, such an implementation may allow bypassing security functionality implemented in a layer. An evaluation needs to carefully analyse such an implementation to ensure that bypassing security functionality in such an implementation is not possible.

4.4 Encapsulation

4.4.1 General

Encapsulation is an architectural principle where objects are "encapsulated" by the functions used for accessing, manipulating or managing the objects. The sum of such functions that encapsulate an object can be viewed as an agent that controls the object.

In such a concept the functions that encapsulate an object can define a security policy for the object. They can define how the object is instantiated, potentially change the object's instantiation (e.g., if the object consists of data, change where and how the data is stored) and also define an access control policy for the object (which requires that the encapsulation function can securely obtain all the information required to make a policy decision like the ensured identity of the accessor).

