

certMILS

White Paper Security Architecture Template

Project reference	731456
Project acronym	H2020-DS-LEIT-2016
Project title	CERT-MILS Compositional security certification for medium- to high-assurance COTS-based systems in environments with emerging threats
Start date:	2017-01-01
Duration	48 months
Programme type	Horizon 2020
Project website	https://certmils.eu/



ezú elektrotechnický
zkušební
ústav


E P O C H E & E S P R I

SYSGO
EMBEDDING INNOVATIONS

THALES


for information security products

TECHNIKON
LECHNIKON

UniControls
Transport and Industrial Control Systems

NXP

Schneider
Electric

**Universität
Rostock**  *Traditio et Innovatio*



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731456.

Editor

Sergey Tverdyshev (SYSGO)

Contributors (ordered according to beneficiary numbers)

Benito Caracuel, Amelia Álvarez (SCHN)

Alvaro Ortega, Jose Emilio Rico (E&E)

Reinhard Hametner (THA)

Sergey Tverdyshev, Holger Blasum (SYSGO)

Tomáš Kertis (UCO)

Thorsten Schulz (UROS)

Disclaimer

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The content of this document reflects only the author’s view – the European Commission is not responsible for any use that may be made of the information it contains. The users use the information at their sole risk and liability.

Executive Summary

The certMILS project (<http://www.certmils.eu/>) aims at easing building and certification of complex critical systems by using a certain architecture for structuring these systems into partitions that run on a separation kernel, called MILS (Multiple Independent Levels of Security / Safety). Once a critical system is structured by use of a separation kernel, then this technical structuring should lend itself also to a similarly logically structured security and safety argument in certification.

Analogous to the separation kernel that is to be used for *building* a MILS system, this white paper provides a security architecture template that is to be used for the *certification* of that MILS system.

The target audience of this document is:

- Developers of systems, based on a MILS architecture, providing them a template about how to describe their MILS system.
- Security evaluators of a MILS-based system, giving hints about how the developer description can be used to argue for compliance to Common Criteria (CC) and IEC 62443.

The assurance case made by the security architecture template in this document identifies as building blocks the security mechanisms implemented by a MILS separation kernel and a typical application payload in partitions and derives typical security architecture arguments for MILS-based systems.



Contents

- Chapter 1 Introduction 1**
- Chapter 2 Template for a simple defence-in-depth design 2**
 - 2.1 Overall system architecture, including partitions 3
 - 2.1.1 Security domain separation 3
 - 2.1.2 List of partitions 3
 - 2.1.2.1 *Partition 1* 3
 - 2.1.2.1.1 List of resources exclusively assigned to partition 1 (RP1) 3
 - 2.1.2.1.2 Interfaces 4
 - 2.1.2.1.3 Application-based security functionality implemented by partition 1 (FP1) 4
 - 2.1.2.2 *Partition 2* 4
 - 2.1.2.2.1 List of resources exclusively assigned to partition 2 (RP2) 4
 - 2.1.2.2.2 Interfaces 4
 - 2.1.2.2.3 Application-based security functionality implemented by partition 2 (FP2) 4
 - 2.1.3 List of resources shared between multiple partitions 5
 - 2.1.4 Security domain separation mechanisms 5
 - 2.2 Initialization / start up 6
 - 2.3 Self-protection and non-bypassability 6
 - 2.3.1 Self-protection 6
 - 2.3.2 Non-bypassability 7
 - 2.4 Application-based security mechanisms 8
 - 2.4.1 Access control 8
 - 2.4.2 Identification and authentication 9
 - 2.4.3 Input validation 9
 - 2.4.4 Secure fallback functionality 9
- Chapter 3 Common Criteria usage of the template 10**
 - 3.1 CC Architecture Work Units 10
 - 3.2 Note on uses of functionality in partitions 12
- Chapter 4 IEC 62443 usage of the template 13**
 - 4.1 MILS separation kernel: conduit or zone? 13
 - 4.2 IEC 62443 [5] Part 4-1 requirements that are well covered by use of MILS separation kernels 13
 - 4.3 IEC 62443 [5] Part 4-2 requirements that are well-covered by the use of MILS systems 15
 - 4.4 Note on uses of functionality in partitions 16
- Chapter 5 Summary and Conclusion 17**
- Chapter 6 List of Abbreviations 18**
- Chapter 7 Bibliography 19**



List of Figures

Figure 1: Simple defence-in-depth design of system M..... 3
Figure 2: A network role has access to a defined number of partitions..... 9

List of Tables

Table 1: Security mechanisms implemented in the partition 2..... 5
Table 2: Mapping of terms the certMILS PP and the security architecture template, including conformity judgments 10
Table 3: IEC 62443 4-1 requirements specifically addressed by MILS systems 13
Table 4: IEC 62443 4-2 requirements specifically addressed by MILS systems 15

Chapter 1 Introduction

Security certification standards (e.g. Common Criteria, IEC 62443) require the architecture description for a product under evaluation/certification. The assumption of this template is that MILS architectures have some features in common, which will re-appear across the security arguments used for certifications of different MILS systems.

The idea is that a system integrator of a MILS system can apply the template in Chapter 2 to derive a security architecture. This template assumes that MILS approach is used (<https://zenodo.org/record/45164>) since it makes mappings/placeholders to/for the corresponding components of a MILS systems (e.g. partition, separation kernel); a worked example is given for a system with two partitions, and it is assumed that it can be easily extended towards larger systems.

Specifically, Chapter 2 is intended to be copy-pasted and filled out by the developer of a MILS system to create the main certification artefact for a Common Criteria (CC) security architecture (in CC terms: this is the security assurance family “development assurance – security architecture”, abbreviated as ADV_ARC) or to be used for IEC 62443. Chapter 3 serves as guidance how to use the developer data provided in Chapter 2 for a CC evaluation. Chapter 4 serves as guidance on how to use the developer data provided in Chapter 2 for an IEC 62443 evaluation.


This document contains the security architecture of a MILS system. A MILS system is a product using a MILS separation kernel. The answer to the threats and the security problem for the MILS system involves the following:

- a. The functionality of the product, which satisfies the security functional requirements according to its deployment;
- b. The design of the product, which in addition resolving the general and security irrelevant aspects, must implement the security functionality under a secure architecture, establishing separate domains for the protection of the assets, a secure start-up and initialization process, and guarantee the non-bypassability and self-protection properties for the architecture.
- c. The techniques, languages and programming tools, which prevent immediate vulnerabilities belonging to the codification practices.

This document presents the mechanisms and design architecture, including the internal architecture that is directly fulfilled by the functionality accessible through the external interfaces of the MILS system.

The approach taken in this document is that security mechanisms implemented by the separation kernel (FSK-x) and partitions (FP1-x and FP2-x) are clearly named so that responsibilities can be easily traced.

Chapter 2 Template for a simple defence-in-depth design

Note: The template in this section is intended to be copy-pasted and adapted by the developer. Depending on the concrete system, not every statement may be applicable, if it is not applicable, just delete it. *Text in italics is more likely to require fine-tuning than text that is not in italics.* Notes (like this text) are written in bold typeface and are to be deleted by the developer while instantiating this template. For easier editing, this PDF contains an attachment with an OOXML (“Microsoft Word”) document containing the template. 

A MILS system typically will have two or more partitions. This template assumes a MILS system with two partitions. Obviously, this template can be extended to a larger number of partitions, if the number of partitions is larger than two.

In this template, without limitation of generality, we assume that partition 1 is of higher or equal criticality as partition 2. You are welcome to replace the terms “partition 1” and “partition 2” by the names for the partitions that you use in your system design.

Typical use cases for two partitions are:

- a firewall, which interfaces networks of two different criticality levels,
- addition of new (less trusted) functionality to a highly trusted existing system
- combination of a trusted small core functionality (e.g. a single custom-developed application) with a less trusted richer functionality e.g. running on an entire operating system like Linux

The template assumes that defence-in-depth is used to shield partition 1 from attackers; that is attackers cannot attack partition 1 directly.

FAQ: Do I have to follow the assignment, selection, refinement suggested below strictly?
Answer: No, a security architecture is not so formal as a PP or ST. That is, you can use this document in any way *you* deem useful to make your security architecture, which includes you can also refine / delete / add text at will.

The overall system design of MILS system M is depicted in Figure 1. Details of Figure 1 will be explained below.

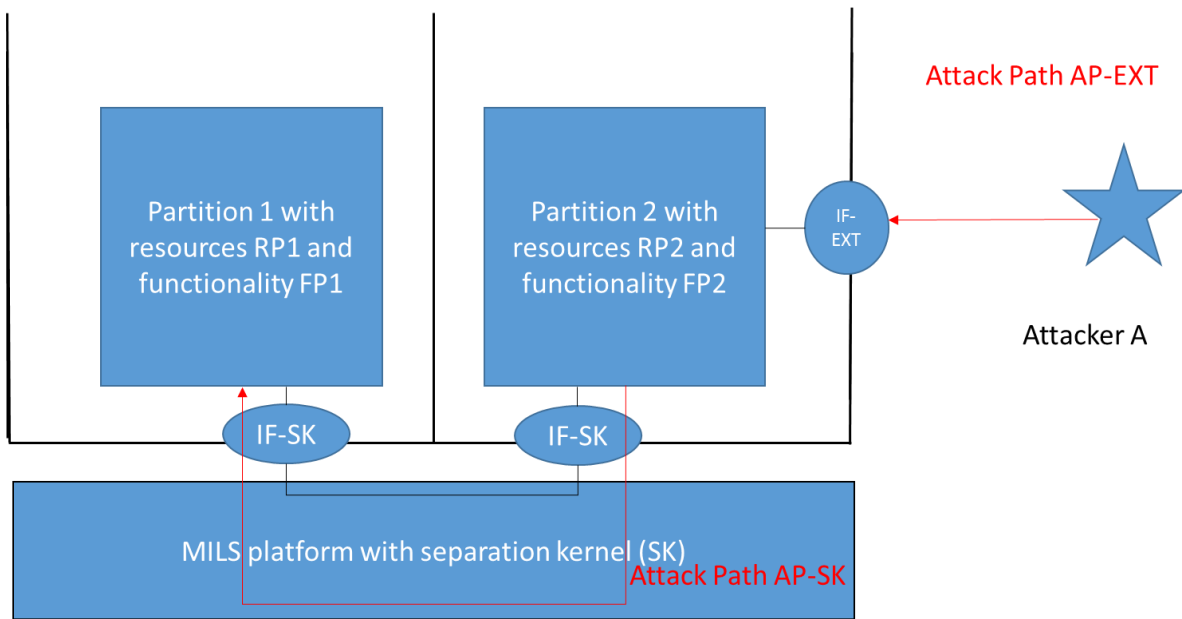


Figure 1: Simple defence-in-depth design of system M

2.1 Overall system architecture, including partitions

This section gives a description of all security elements of the MILS system, which implements the security functionality.

2.1.1 Security domain separation

This section describes the functionality of security domain separation provided by the separation kernel.

Note: Our running example with just two partitions is quite simple. If there are many partitions of the same type, that is each partition the same security policy applied to it, then it also can make sense to group on partitions of the same type together, e.g. by referring to them as “partitions 3 to 7” or the “XXX partitions”.

Each security domain is a partition maintained by the separation kernel.

2.1.2 List of partitions

This section is based on the configuration by the system integrator.

The partitions are:

- partition 1, which [short one-line characterization], and,
- partition 2, which [short one-line characterization].

2.1.2.1 Partition 1

2.1.2.1.1 List of resources exclusively assigned to partition 1 (RP1)

RAM is exclusively assigned to the partition as configured by the integrator.

Note: the concept of CPU time is explained in certMILS D2.1 (Base PP) [1] and the “CPU time modules” in certMILS D2.2 (List of extensions of the base PP) [2]. In the selection below you also can select any combination of methods, e.g. “assignment of time slices and exclusive CPU affinity of partition 1”.

CPU time is exclusively assigned to the partition in the following way:



[assignment of time slices when period-based scheduling is used | maximum priority when priority-based scheduling is used | CPU affinity of the partition when CPU affinity-based CPU allocation is used]

The following other resources are exclusively assigned to the partition: [...]

2.1.2.1.2 Interfaces

The following interfaces are identified:

- Partition 1 communicates with partition 2 via interface IF-SK.

2.1.2.1.3 Application-based security functionality implemented by partition 1 (FP1)

This subsection describes security functionality implemented by application developers of applications within the partitions.

Note: Security functionality implemented within applications supplements the overall domain separation provided by the separation kernel (Sections 2.1.1 to 2.1.3).

As an example, the following table may summarise the observable security services mechanisms. This example can be followed by the users of the security template. Note that the security mechanisms in this example are further described in Section 2.4 of this document.

The following table lists the security mechanisms implemented by applications in partition 1.

Table 1: Security mechanisms implemented in partition 1

Allocation to the partition	Name	Security Mechanism
	FP1-FB	Fallback functionality (See Section 2.4.4)
	FP1-IV	Input Validation (See Section 2.4.3)

2.1.2.2 Partition 2

2.1.2.2.1 List of resources exclusively assigned to partition 2 (RP2)

RAM is exclusively assigned to the partition as configured by the integrator.

CPU time is exclusively assigned to the partition in the following way:

[assignment of time slices when period-based scheduling is used | maximum priority when priority-based scheduling is used | CPU affinity the partition when CPU affinity-based CPU allocation is used]

The following other resources are exclusively assigned to the partition: [...]

2.1.2.2.2 Interfaces

The following interfaces are identified:

- Partition 2 communicates with partition 1 via interface IF-SK.
- Partition 2 is exposed to attacker A via interface IF-EXT (attack path AP-EXT).

2.1.2.2.3 Application-based security functionality implemented by partition 2 (FP2)

The following table lists the security mechanisms implemented by applications in partition 2.

Table 1: Security mechanisms implemented in the partition 2

Allocation to the partition	NAME	Security Mechanisms
	FP2-AC	Access control policy (See Section 2.4.1)
	FP2-UA	User authentication (See Section 2.4.2)
	FP2-IV	Input Validation (See Section 2.4.3)

2.1.3 List of resources shared between multiple partitions

The following resources are shared between partition 1 and partition 2:

[E.g. shared files, shared memory, etc., if any. This list also can be empty.]

2.1.4 Security domain separation mechanisms

Note: Security domain separation relies on the separation kernel. Hence there is no additional work for the MILS system user in this subsection.

The separation kernel provides the following security mechanisms:

FSK-MEM: The separation kernel ensures that memory is only accessible to the partition(s) as it has been assigned by the system integrator.

FSK-TIME: The separation kernel ensures that CPU time is only available to the partition(s) as it has been assigned by the system integrator. This can be done by assigning period- or priority-based scheduling, or assigning whole cores. Also, any combination of the methods can be applied.

FSK-RES: The separation kernel ensures that also each other resource is only available to the partition(s) as it has been assigned by the system integrator.

FSK-COM: All communication between partitions via the interface IF-SK is mediated by the separation kernel. When communication is not explicitly configured, the default is that any communication via the IF-SK interface is denied.

FSK-IS: The separation kernel controls the CPU instruction set available to applications, so that applications cannot use CPU instructions that could be used to arbitrarily reconfigure access to memory.

In particular:

- Memory access of applications in partition *n* are confined to partition *n*.
- Resources allocated to partition *n* are made available to partition *n* only.

Partitions are separated by the separation kernel. The separation kernel controls all accesses to memory, hence confidentiality and integrity of the resources of each partition, as well as the integrity of the functionality of each partition are ensured, except for memory that has been explicitly shared (Section 2.1.3) by the system integrator.

The use of the separation kernel communication services at interface IF-SK is secure, because the applications in each partition do check all arguments coming via IF-SK. Neither the attacker A nor applications in any other partition can bypass the communication mechanisms of the separation kernel.

2.2 Initialization / start up

After power-on, the following initialization chain is used:

The first step that happens after power-on is that the boot-loader is executing.

[Only write this paragraph, picking, at your choice, one or more of the bullets, if applicable for your system] During this stage:

- *The boot-loader checks the authenticity of the image by ...*
- *The boot-loader checks the integrity of the image by ...*

The boot-loader loads the separation kernel. While the bootloader is running, the separation kernel has not started and there are no active partitions. Interfaces IF-SK and IF-EXT are not available.

The separation kernel now boots and allocates resources to the partitions. Only when the separation kernel has completely started, it runs partitions and interface IF-SK is available to its partitions. It is from now on, but not before, that the functionalities provided by the partitions, FP1 and FP2, are available. Then, after complete initialization, the interface IF-EXT (realized by FP2 of partition 2) to system M is available. It is not possible to access to partitions via IF-SK or other means the boot-loader. As all communication via IF-EXT is under control of partition 2 that only can communicate via IF-SK with the separation kernel, it is also not possible to access the boot-loader via IF-EXT.

2.3 Self-protection and non-bypassability

2.3.1 Self-protection

Note: Self-protection, including defence-in-depth, is partially provided by the separation kernel and supplemented by application-based checks.

At interface IF-EXT, self-protection is exercised by functionality FP2 in partition 2 checking against illegal accesses from attacker A.

The following external interfaces are provided by IF-EXT:

[list of functions / sub-interfaces in IF-EXT]

System M is protected from tampering over the network, given that all the accesses to M via the external interfaces are authenticated (user identification and authentication mechanism FP2-UA implemented in the applications running in partition 2) and authorized (Access control policy mechanism FP2-AC running in partition 2). Moreover, all the inputs through this interface are validated to avoid problems processing the inputs in the MILS system (FP2-IV). IF-EXT is protected from physical attacks *[by a tamper detection mechanism | by physical protection against attackers]*. Therefore, it is considered that this IF-EXT is protected from manipulation from external entities that may result in changes to the security functionality implemented in partition 2.

Denial-of-service attacks on IF-EXT are mitigated by deterministic scheduling of time by the separation kernel between partition 1 and partition 2 by allocation of CPU time (FSK-TIME; configuration as described in Section 2.1.4). That is, even when partition 2 is flooded with network traffic, time partitioning ensures that partition 1 *[pick the one applicable: "is always available" | "still gets cyclically scheduled" | "cannot be blocked by partition 2 during its higher scheduling priority"]*.

As additional layer of defence, if partition 2 is compromised, then the separation kernel and its configuration restrict any harm that partition 2 can do towards partition 1 via the interface IF-SK, as all interactions in IF-SK have to be explicitly configured (default: deny-all).

On the hardware level, the self-protection depends on system M running on a hardware platform with memory management and different CPU privilege modes, where:

- the separation kernel configures memory so that applications are confined to certain memory regions (FSK-MEM),

- the separation kernel restricts the instruction set so that applications cannot use the most privileged instructions to arbitrarily reconfigure access to memory (FSK-IS).

This use of memory management and CPU privilege modes by the separation kernel to establish security domains follows the approach described by Saltzer and Schroeder (“The Protection of Information in Computer Systems,” *Proc. IEEE*, vol. 63, no. 9, pp. 1278-1308, 1975, <http://web.mit.edu/Saltzer/www/publications/protection/>).

2.3.2 Non-bypassability

The property shall ensure that no available interface can be used to bypass the MILS system. This means that every available interface must be either unrelated to the security functions that are claimed (and does not interact with anything that is used to satisfy the security functions) or else the absence of interaction is described in other development evidence (e.g., modular design of the application).

Note: There are different possible argument styles to do this, which also can be mixed.

Resource-focused arguments: For each resource protected by the MILS system (e.g. cryptographic keys) or security function defined in Section 2.1.2, a rationale shall be included explaining how the access to security mechanisms cannot be bypassed.

Therefore, the following approach could be followed:

1. Define the resources protected by MILS and the security functionality provided (Use design documentation)
2. Describe the authorized method to access to each one of the resources or separation kernel services
3. Include a rationale justifying how the only way to access to the resource or separation kernel services is the authorized one and ensuring that no available interface can be used to bypass.

For example:

- 1- The resources protected are: the data in partition 2 (e.g. cryptographic keys, defined as assets in Section 2.1.2.2.1) and the security functionality in partition 2 (Section 2.1.2.2.3) available through Interface IF-EXT.
- 2- To access to the cryptographic keys, it is required to call the API call GetKey(). To call this API function, it is necessary to be authenticated and authorized by the MILS system (FP2-UA).
- 3- It is considered that the resource “cryptographic keys” is only accessible through the security function (FP2-AC, API call GetKey). To call this API function, the user must be authenticated and authorized by the MILS system. No other way to access to this resource is available in the MILS system. Therefore, it is not possible to bypass the authorized mechanism in the MILS system.

Interface-focused arguments: Alternatively, it is possible that a set of resources and security functions are accessed in the same way. In those cases, the same rationale may apply. Thus it is possible to make the same argument on interfaces of the partitions. This style can be economic when there are multiple resources and security functions but not that many interfaces.

The example below first identifies the resources (resource-focused argument) and then explains the non-bypassability of IF-SK by IF-EXT (interface-focused argument for FP1 and RP1). The argument uses security mechanisms implemented by the separation kernel and individual partitions.

The functionalities and resources protected by the MILS system and the separation kernel are:

- FP1 (Section 2.1.2.1.3), RP1 (Section 2.1.2.1.1)

- FP2 (Section 2.1.2.2.3), RP2 (Section 2.1.2.2.1)

The separation kernel's memory separation (FSK-MEM) forbids memory accesses to RP1 that would be bypassing FP1. The applications in FP1 have been designed to validate the data coming from partition 2 (FP1-IV). Hence partition 2 cannot interfere with the resources of partition 1 via the IF-SK interface. Data and resources in RP1 are only exposed via FP1 (FSK-COM) and also cannot be bypassed by abuse of CPU privilege mode (FSK-IS). Hence, there is no direct interface from the attacker via IF-EXT to FP1 and RP1. The separation kernel provides enough resources to partition 1 by FSK-MEM and FSK-TIME. FP1-FB ensures that partition 1 also can run without input from partition 2. Thus, the fall-back functionality of partition 1 is ensured by the implementation of FP1-FB in combination with the protection of partition 1 provided by the above-mentioned separation kernel mechanisms.

Interactions via shared resources (Section 2.1.3) do not lead to bypass, because there *[there are no shared resources | argument why possible interaction via resource sharing does not constitute bypass]*.

Privilege levels in the external network are mapped to roles in the separation kernel. FP2-UA ensures that users are identified and FP2-IV ensures that inputs from users are checked for validity.

2.4 Application-based security mechanisms

In the following is a typical list of security mechanisms encountered in *applications* of MILS systems. Whether these occur in the applications of *your* system, will depend on your needs. So optionally, you can choose to use describe here the security mechanisms that are applicable to your system, however the description might need adaptation. To have a list of security mechanism that is referred especially is useful if you use the same security mechanism from different partitions. Alternatively, you can also describe security mechanisms in Section 2.1.2 and decide not to use this section at all.

This section describes the security mechanisms available in the system M, which are provided by applications.

2.4.1 Access control

In a MILS system with a separation kernel, the separation kernel will manage some hardware component, e.g. an embedded system. The embedded system itself is likely to be connected to one or several networks. A MILS system allows, for each role R in the network, to restrict its access to certain partitions (an example is given in Section 2.4.2). When there is more than one role in the network, then an application in a partition will have some knowledge of the network roles.

FP2-AC: The application in partition 2 restricts access to the network to the partitions with role **XXX**.

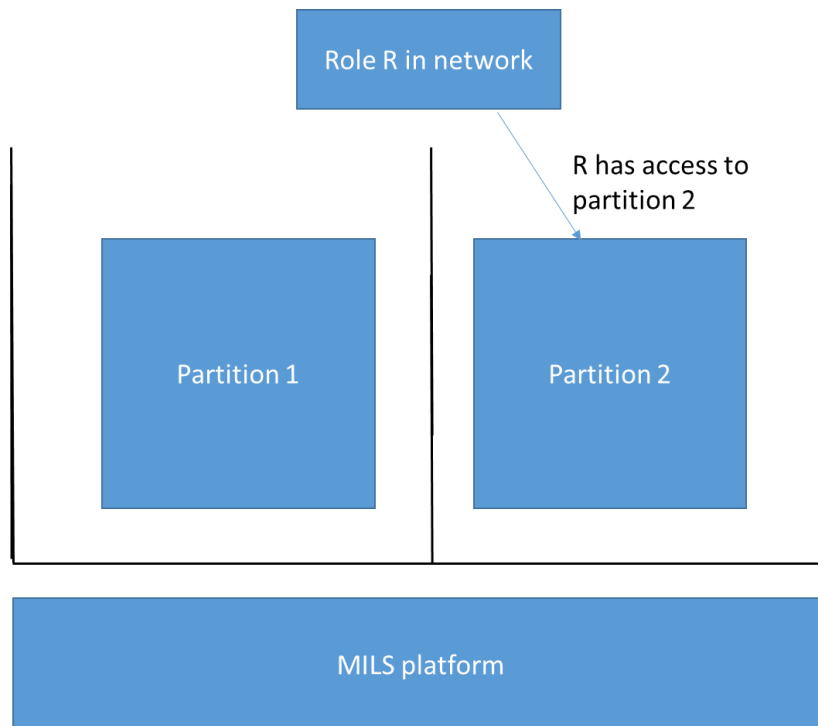


Figure 2: A network role has access to a defined number of partitions

2.4.2 Identification and authentication

For maintaining the access control described in Section 2.4.1, probably some identification and authentication of users wanting to come in via the network will be used (e.g., pass-phrase or certificate-based).

FP2-UA: The application in partition 2 uses the following authentication for users that come in via the network: **XXX**

2.4.3 Input validation

If input is accepted from users via some interface, then this input will be probably verified.

FP1-IV: The application in partition 1 verifies user input in the following way: [...]

FP2-IV: The application in partition 2 verifies user input in the following way: [...]

2.4.4 Secure fallback functionality

In case a part of the system gets compromised, there might be a secure fallback functionality. The main part of the fallback functionality is already provided by the separation kernel, if the system integrator that configures that the fallback partition has access to CPU cycles. Hence there is little or even nothing to implement for the application developer who obtains a MILS system from a third party to ensure that the secure fallback is not compromised. In some cases, there might be some need to for the fallback application to detect that the other partition fails, which would be encoded into the application.

FP1-FB: The application in partition 1 does not depend on input or actions from partition 2 when it serves as secure fallback.



Chapter 3 Common Criteria usage of the template

Texts in this chapter could be used as a starting point by a Common Criteria for Information Technology Evaluation (CC, [3]) evaluator for use in her CC Evaluation Technical Report (ETR) on the CC development assurance class (ADV), mainly for the development assurance’s software architecture class (ADV_ARC).

3.1 CC Architecture Work Units

Table 2 below describes how the certMILS PP [1] and the security architecture templates (Chapter 2 above) match to a CC evaluation. In the CC texts the following terms will occur:

- Target of evaluation (TOE), which in our case is the MILS system, possibly accompanied by guidance.
- TOE security functionality (TSF), which in our case is the functionality of the MILS system that must be relied upon for the correct enforcement of its security claims. Note that from the point of view of certifying the MILS system, a particular TSF can be provided either by the separation kernel or its application load.
-

Table 2: Mapping of terms the certMILS PP and the security architecture template, including conformity judgments

CC CEM work unit	Relevant CC and CEM guidance <i>Italics</i> are used for CC/CEM quotations; plain text is used for our interpretation. The bold markup of is used for particularly relevant keywords (our interpretation).	Conformity evidence according to the sections of this document	Evaluator judgment
ADV_TDS.1-1	According to [3], CEM, ADV_TDS.1-1, <i>“The evaluator shall examine the TOE design to determine that the structure of the entire TOE is described in terms of subsystems.”</i> Subsystems must be identified.	Section 2.1	The developer has identified each partition of the separation kernel with a CC subsystem .
ADV_ARC.1-2	According to [3], CEM, paragraphs 576-577, <i>“security domains refer to environments supplied by the TSF for use by potentially-harmful entities; for example, a typical secure operating system supplies a set of resources (address space, per-process environment variables) for use by processes with limited access rights and security properties. The description of the security domains shall take into account all of the SFRs claimed by the TOE.</i> <i>For some TOEs such domains do not exist because all of the interactions available to users are severely constrained by the TSF. A packet-filter firewall is an example of such a TOE.</i>	Section 2.1	The separation kernel is compliant to [1] and guarantees for each partition the confidentiality (OT.CONFIDENTIALITY in [1]) and integrity (OT.INTEGRITY in [1]) of its resources. The use of the separation kernel for domain separation of the MILS system is discussed in the security architecture, section “Security domain separation”, which lists the security

CC CEM work unit	Relevant CC and CEM guidance <i>Italics</i> are used for CC/CEM quotations; plain text is used for our interpretation. The bold markup of is used for particularly relevant keywords (our interpretation).	Conformity evidence according to the sections of this document	Evaluator judgment
	<p><i>Users on the LAN or WAN do not interact with the TOE, so there need be no security domains; there are only data structures maintained by the TSF to keep the users' packets separated.</i>"</p> <p>According to [3], Part 3, Annex A.1.1, Paragraph 535, "domain separation is a property whereby the TSF creates separate security domains for each untrusted active entity to operate on its resource, and then keeps those domains separated from one another so that no entity can run in the domain of any other." The TOE uses a separation kernel which complies to the Base MILS Platform Protection profile[1].</p>		domains of the MILS system, and how the MILS system security domains are mapped to partitions in the separation kernel.
ADV_ARC.1-3	<p>According to [3], CEM, paragraphs 578-580 "<i>the information provided in the security architecture description relating to TSF initialization is directed at the TOE components that are involved in bringing the TSF into an initial secure state (i.e. when all parts of the TSF are operational) when power-on or a reset is applied.</i></p> <p><i>This discussion in the security architecture description should list the system initialization components and the processing that occurs in transitioning from the "down" state to the initial secure state.</i></p> <p><i>It is often the case that the components that perform this initialisation function are not accessible after the secure state is achieved; if this is the case then the security architecture description identifies the components and explains how they are not reachable by untrusted entities after the TSF has been established. In this respect, the property that needs to be preserved is that these components either 1) cannot be accessed by untrusted entities after the secure state is achieved, or 2) if they provide interfaces to untrusted entities, these TSFI cannot be used to tamper with the TSF."</i></p>	Section 2.2	The developer has described in section "Initialization / start up" of the security architecture the initialization chain including boot loader, separation kernel and applications. The separation kernel only will allow applications to run once the initialization of all partitions has finished. Untrusted entities cannot access components that perform initialisation after the initial secure state is reached.

CC CEM work unit	Relevant CC and CEM guidance <i>Italics</i> are used for CC/CEM quotations; plain text is used for our interpretation. The bold markup of is used for particularly relevant keywords (our interpretation).	Conformity evidence according to the sections of this document	Evaluator judgment
ADV_ARC.1-4	According to [3], Part 3, Annex A.1.1, paragraph 533 “ Self-protection refers to the ability of the TSF to protect itself from manipulation from external entities that may result in changes to the TSF. Without these properties, the TSF might be disabled from performing its security services”. Self-protection only applies to the services provided by the TSF through the interfaces of the TSF, hence it covers input user data handling, preventing tampering of the TSF.	Section 2.3.1	Self-protection at interface IF-EXT is provided at application level, as described in Section “Self-protection” of the security architecture. Self-protection at interface IF-SK is provided by the separation kernel, as described in Section “Partition 2: Application-based security functionality” of the security architecture.
ADV_ARC.1-5	According to [3], Part 3, Annex A.1.2, paragraph 537 “ Non-bypassability is a property that the security functionality of the TSF is always invoked and cannot be circumvented when appropriate for that specific mechanism”. We give a rationale shall be included for each one of the TSF’s interfaces explaining how the TOE’s security mechanisms protect the TOE from tampering through that interface.	Section 2.3.2	The developer has described the implementation of non-bypassability in Section “Non-bypassability” of the security architecture. The developer has provided a rationale for each one of the TSF’s interfaces .

3.2 Note on uses of functionality in partitions

The previous Section 3.1 mostly gave guidance how to satisfy CC work units as architectural properties. It should be remarked that the functionality provided in the partitions would be the source of security functional requirements. For instance, the example partition functionality in Section 2.1.2 could probably be used as follows:

- FP1-FB → Protection of the TSF CC class (FPT), in particular failure with preservation of secure state (FPT_FLS in part 2 of [3])
- FP1-AC → User data protection CC class (FDP), in particular access control policy (FDP_ACC) and access control functions (FDP_ACF in part 2 of [3])
- FP2-UA → Management CC class (FMT), in particular user identification and authentication (FIA_UID, FIA_UAU) and possible security management roles (FMT_SMR in part 2 of [3])

As this document focuses on the security architecture and moreover partition load is highly flexible, we do not expand the mapping of partition load to SFRs any longer here.

Chapter 4 IEC 62443 usage of the template

If a MILS system integrator uses in the security architecture templates in Section Chapter 2 to build an assurance case, then the assurance is evidence provided by the applicant “in support of the capabilities that are intended to demonstrate compliance to the selected requirement(s)” [4], further called “supporting evidence” in the tables of this section.

Texts in this chapter could be used by an IEC evaluator for use as results in an evaluation according to IEC 62443 [5] Part 4-1, are marked with “result – remarks” in the table of this section..

Note: IEC 62443 is still under development. Hence we refer to the following draft document versions:

- Part 1.1: Draft 6, Edit 4, March 2017;
- Part 4.1: Draft 3, Edit 11, November 2016;
- Part 4.2: Draft 4, Edit 1, January 2017.

4.1 MILS separation kernel: conduit or zone?

Due to its strong separation properties, a MILS system often is used similar to a firewall. An IEC 62443 evaluation divides a design into zones, which are connected with each other by conduits. It is the choice of the developer whether to put a firewall into a conduit as e.g. in [5], Part 1.1 (Draft 6, Edit 4, March 2017), Annex B.3 or into a zone as e.g. [5], Part 1.1, Section 9.4 or in [6] Section 8, zone “Corporate Boundary Management”. In fact, the development of such a design can undergo multiple iterations [6] and a firewall / separation kernel could be put into a conduit at a coarser level and assigned to a zone (or to its own zone at a finer level). Regardless whether a MILS separation kernel is used as conduit between zones or its own zone, the partitions in a MILS separation kernel, in some use cases, could be (belonging to or representing) different IEC 62443 zones.

4.2 IEC 62443 [5] Part 4-1 requirements that are well covered by use of MILS separation kernels

Table 3 below gives some IEC 62443-4-1 requirements that are specifically addressed by MILS separation kernels, without taking into account functionality provided by the applications. As of now, Table 3 is mainly meant to be used as starting point. These requirements were based on a previous exploratory analysis of IEC 62443-4-1 (Draft 3, Edit 11, November 2016) by certMILS partners [7] and are expected to evolve during further work in certMILS.

Table 3: IEC 62443 4-1 requirements specifically addressed by MILS systems

IEC 62443-4-1 ID	IEC 62443-4-1 Requirement <i>Italics</i> are used for quotations. The bold markup of is used for particularly relevant keywords (our interpretation).	Supporting evidence according to the sections of this document	Result – remarks
SR-2	“ <i>All products shall have an up-to-date threat model with the following characteristics:</i> a) <i>correct flow of categorized information throughout the system;</i> x) <i>trust boundaries;</i> y) <i>processes;</i>	Section 2.1	The developer has provided a threat model in Section “Security domain separation” of the security architecture. Trust boundaries are IF-SK and IF-EXT.



IEC 62443-4-1 ID	IEC 62443-4-1 Requirement <i>Italics</i> are used for quotations. The bold markup of is used for particularly relevant keywords (our interpretation).	Supporting evidence according to the sections of this document	Result – remarks
	z) <i>data stores</i> ; aa) interacting external entities ; [...].”		Processes are the applications assigned to the partitions. Each data flow is indicated by an explicit communication channel. Interacting external entities are listed along with interfaces in Section “Non-bypassability” of the security architecture.
SD-1	“A process shall be employed for developing and documenting a secure design that identifies and characterizes each exposed interface of the product, including physical and logical interfaces, to include: <i>an indication of whether the exposed interface is externally accessible (by other products), or internally accessible (by other components of the product) , or both;</i> [...] rr) a determination of whether access to the exposed interface crosses a trust boundary; [...].”	Section 2.1	In the secure design provided by the developer (Sections “Security domain separation” and “Non-bypassability”), each exposed interface , including trust boundaries , in the form of partitions, of the product is identified and it is defined whether the interfaces is externally or internally accessible..
SD-2	“A process shall be employed for including multiple layers of defense where each layer provides additional defense mechanisms. Each layer should assume that the layer in front of it may be compromised. Secure design principles are applied to each layer. A process shall be employed for assigning responsibilities to each layer of defense reducing the attack surface of the inner layers.”	Chapter 2	As a process for including an additional layer of defense , the developer has used a MILS separation kernel. The developer has presented a layered defense-in-depth design is described in the Section “Security domain separation”, “Initialization / start up”, “Self-protection and non-bypassability”.
SD-6	“A process shall be employed to ensure that security industry recommended practices are documented and applied to the design process. Industry recommended practices shall be periodically reviewed and updated. Secure design industry recommended practices include but are not be limited to:	Section 2.1	The developer has used a separation kernel as proven secure component, with economy of mechanism (striving for simple design). Use of a separation kernel allows to establish least privilege , is established proven secure design pattern



IEC 62443-4-1 ID	IEC 62443-4-1 Requirement <i>Italics</i> are used for quotations. The bold markup of is used for particularly relevant keywords (our interpretation).	Supporting evidence according to the sections of this document	Result – remarks
	a) least privilege (<i>granting only the privileges to users/software necessary to perform intended operations</i>); b) using proven secure components/designs where possible; c) economy of mechanism (<i>striving for simple designs</i>); d) using secure design patterns ; e) attack surface reduction ; f) all trust boundaries are documented as part of the design [...]		[8] that allows attack surface reduction. Using a separation kernel naturally lends to document all trust boundaries (see Section “Non-bypassability” of the security architecture).

4.3 IEC 62443 [5] Part 4-2 requirements that are well-covered by the use of MILS systems

Table 4 below gives some IEC 62443-4-2 requirements that are specifically addressed by MILS separation kernels, without taking into account functionality provided by the applications. As of now, Table 4 is mainly meant to be used as starting point. These requirements were based on a preliminary analysis of IEC 62443-4-2 (Draft 4, Edit 1, January 2017) [7] and are expected to evolve during further work in certMILS.

Table 4: IEC 62443 4-2 requirements specifically addressed by MILS systems

IEC 62443-4-2 ID	IEC 62443-4-2 Requirement <i>Italics</i> are used for quotations. The bold markup of is used for particularly relevant keywords (our interpretation).	Supporting evidence according to the sections of this document	Result – remarks
CR 5.1	“ <i>Components shall support a segmented network as defined in ISA 62443-3-2, as needed, to support the broader network architecture based on logical segmentation and criticality.</i> ”	Section 2.1.4 (FSK-COM)	The developer used a separation kernel so that all communication between partitions at interface IF-SK is mediated by the separation kernel, that is the logical segmentation of the network is mirrored by the separation kernel’s domain separation and communication policy.
CR 7.1	“ <i>Components shall provide the capability to maintain essential functions in a degraded mode during a DoS event.</i> ”	Section 2.1	The developer ensures allocation of CPU time to ensure to be able to maintain essential functions in a degraded mode during a DoS event as described in Section “Security domain



IEC 62443-4-2 ID	IEC 62443-4-2 Requirement <i>Italics</i> are used for quotations. The bold markup of is used for particularly relevant keywords (our interpretation).	Supporting evidence according to the sections of this document	Result – remarks
			separation” of its security architecture. In the degraded mode, even if partition 2 is under attack from attacker A, partition 1 will run normally.
CR 7.2	“ <i>Components shall provide the capability to limit the use of resources by security functions to prevent resource exhaustion.</i> ”	Section 2.1	The developer ensures allocation of CPU time to prevent resource exhaustion as described in Section “Security domain separation” of its security architecture”..

4.4 Note on uses of functionality in partitions

The previous Sections 4.2 and 4.3 mostly gave guidance how to satisfy IEC 62443 work units which exploit architectural properties of the separation kernel. It should be remarked that the functionality provided in the partitions would be the source of security functional requirements. For instance, the example partition functionality in Section 2.4 could probably be used for IEC 62443 Part 4-2 foundational requirements as follows:

- FP1-IV, FP2-IV, FP1-FB ->system integrity (FR-3 in IEC 62443 Part 4-2)
- FP1-AC ->data confidentiality (FR-4 in IEC 62443 Part 4-2)
- FP2-UA ->identification and authentication (FR-1 in IEC 62443 Part 4-2)

As this document focuses on the security architecture and moreover partition load is highly flexible, we do not expand the mapping of partition load to foundational requirements any longer here.

Chapter 5 Summary and Conclusion

We have claimed before that “once a critical system is structured by use of a separation kernel, then this technical structuring should lend itself also to a similarly logically structured security and safety argument in certification.” We have described the properties of a separation kernel and have generated an assurance case that use these properties. For a typical MILS two-partition scenario this Security Architecture Template describes a developer could build an assurance case developer and gives advice down to the level of the building blocks on how to use the developer description as evidence in a CC and an IEC 62443 evaluation.

The separation kernel provides clear domain separation, trust boundaries, information flows and defense-in-depth. A critical system that uses a separation kernel will of course use the separation kernel to protect its assets, including the security and safety functionality it provides itself. In this place a limitation of the template-based approach comes in: while the separation kernel itself is a well-defined system, it is much harder to reason on “typical” systems using the separation kernel. We have worked into this direction, by identifying “typical” properties of applications (such as input validation, or authentication of users). Yet a user of the template cannot be discharged from the responsibility to check how her own system’s applications functionality matches the templates, and to adapt these to her own needs. In this respect, the security architecture templates are less out-of-the-box usable than the certMILS separation kernel base PP and its modules, that indeed are intended to be used by “filling in” the instantiations. Perhaps in future, more research on “typical” separation kernel applications and their functionalities could even help to establish certain separation kernel usage design patterns. These design patterns then could possibly be used to build a modular security architecture template for MILS systems, that even covers common application loads, akin to the certMILS modular PP, that covers common separation kernel extensions - but this is beyond the scope of this work.

Nonetheless, giving templates for a separation kernel with a specific example partition load, as pursued by this white paper, should help in figuring out how to write the more separation-kernel-dependent parts of a security architecture for a system that uses a separation kernel. An encouraging finding was that it was comparatively doable to identify where some CC and IEC 62443 credits could be obtained. Moreover, during certMILS the template already has been successfully be used to describe its own security architecture.

Chapter 6 List of Abbreviations

Abbreviation	Translation
ADV	Development Assurance
ADV_ARC	Development Assurance – Software Architecture
API	Application Programming Interface
CC	Common Criteria for Information Technology Security
CEM	Common Evaluation Methodology
CPU	Central Processing Unit
DoS	Denial-of-Service
ETR	Evaluation Technical Report
FDP	User data protection CC class
FMT	Security Management CC class
FSK	Functionality implemented by Separation Kernel
IEC	International Electrotechnical Commission
MILS	Multiple Independent Levels of Safety / Security
PP	Protection Profile
SFR	Security Functional Requirement
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TOE Security Functionality Interface

Chapter 7 Bibliography

- [1] certMILS, “D2.1: Base MILS Protection Profile,” 2018.
- [2] certMILS, D2.2: List of extensions of base PP, 2018.
- [3] Common Criteria Sponsoring Organizations, “Common Criteria for Information Technology Security Evaluation. Version 3.1, revision 5,” April 2017. [Online]. Available: <http://www.commoncriteriaportal.org/cc/>.
- [4] IEC, “OD2061: IEC System of Conformity Assessment Schemes for Electrotechnical Equipment and Components (IECEE System): Industrial Cyber Security Program,” 2016. [Online]. Available: <https://www.iecee.org/documents/refdocs/downloads/od-2061ed.1.0.pdf>.
- [5] International Electrotechnical Commission, Technical Committee 65: Industrial-process measurement and control, “IEC 62443: Security for industrial automation and control systems,” 2017. [Online]. Available: <http://www.isa99.org/>.
- [6] Tofino Security, “White Paper: Using ISA/IEC 62443 Standards to Improve Control System Security,” 2014. [Online]. Available: [https://www.tofinosecurity.com/sites/default/files/common/white-papers/Using-ISA_IEC-62443-Standards-WP-v1.2%20\(May%202014\).pdf](https://www.tofinosecurity.com/sites/default/files/common/white-papers/Using-ISA_IEC-62443-Standards-WP-v1.2%20(May%202014).pdf).
- [7] S. Nordhoff and H. Blasum, “Ease Standard Compliance by Technical Means via MILS,” 2017. [Online]. Available: <https://zenodo.org/record/571175>.
- [8] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad and M. Stal, Pattern-Oriented Software Architecture, A System of Patterns, John Wiley & Sons, 1996.
- [9] certMILS, D2.4; Guidelines to use and apply PP for all involved stakeholders, 2018.