Architecture and composition in security standards

Webinar: Cybersecurity standards and certification - the challenges

05 September 2018

https://www.cyberwatching.eu/free-webinar-cybersecurity-standards-and-certification-challenges

Holger Blasum, SYSGO

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

Security, architecture and composition (1)

Architecture used through-out

- Threads
  - Within a program
  - to keep computing agents separate
    - threads are cooperative agents
- Address spaces
  - Provided by OS
  - to keep programs separate (“safety”)
    - Typically resources are shared (e.g. allocatable RAM, CPU time)
- Secure execution environments
  - Provided by hypervisor/MILS OS[*]/separation kernels
  - Virtual environment with
    - full resource virtualization
    - controlled communication

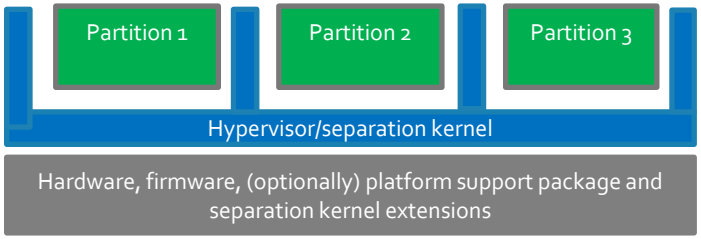
Known limitations

- Low-level mechanisms can bypass high-level mechanisms
  - E.g. OS access control is useless if you can access (unencrypted) harddisk and read it out bitwise
  - E.g. side channels can measured when a memory controller is shared (e.g. Kuzhiyelil 2015)
  - E.g. out-of-order execution (Meltdown, Spectre)
- Composition can bypass component-wise security design
  - Think of a component on system A making sound signals via a loudspeaker to a component with a microphone on system B
  - Example where behavior of B depends on input of A allows to infer size of data: McCullough 1988

[*] **MILS** = Multiple Independent Levels of Security / Safety (https://en.wikipedia.org/wiki/Multiple_Independent_Levels_of_Security)
Don **Kuzhiyelil**, Sergey Tverdyshev, A Secure Update Architecture for High Assurance Mixed-Criticality System, 2015, https://www.escar.info/images/Datastore/2015_escar_EU/16_Kuzhiyelil_escar_EU_2015.pdf (free registration required).)
Daryl **McCullough**, Noninterference and the Composability of Security Properties, Proceedings of the 1988 IEEE conference on Security and privacy, SP'88, p. 177-186, 1988, IEEE Computer Society, Washington, DC, USA, http://dl.acm.org/citation.cfm?id=1949221.

Security, architecture and composition (2)

- No architecture also not an option
  - Building systems from components the norm in
    - Humans constructing buildings and systems
    - Organisms
- Use security design patterns
  - Thread, address space, hypervisor
  - Use COTS components
- certMILS
  - uses separation kernel for three demonstrators; smart grid railway; subway
  - certMILS produces a security architecture for each system
    - What are the security domains and their boundaries
    - What could go wrong at these boundaries
      - Non-bypassability argument
    - certMILS will provide generic Security Architecture Templates for analyzing systems using a separation kernel
      - for Common Criteria for information technology security (CC) and IEC 62443 contexts



E.g. Common Criteria (CC) - Security Architecture

ADV_ARC.1.2C ***The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs.***

ADV_ARC.1-2 The evaluator ***shall examine*** the security architecture description to determine that it describes the security domains maintained by the TSF.

ADV_ARC.1.4C ***The security architecture description shall demonstrate that the TSF protects itself from tampering.***

ADV_ARC.1-4 The evaluator ***shall examine*** the security architecture description to determine that it contains information sufficient to support a determination that the TSF is able to protect itself from tampering by untrusted active entities.

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. For TOEs that have dependencies on other IT entities, it is often the case that the TOE uses services supplied by the other IT entities in order to perform its functions. In such cases, the TSF alone does not protect itself because it depends on the other IT entities to provide some of the protection. For the purposes of the security architecture description, the notion of *self-protection* applies only to the services provided by the TSF through its TSFI, and not to services provided by underlying IT entities that it uses.

Security Domains = separation kernel partition

Bypassing and tampering can be addressed by separation kernel concepts

ADV_ARC.1.5C ***The security architecture description shall demonstrate that the TSF prevents bypass of the SFR-enforcing functionality.***

ADV_ARC.1-5 The evaluator ***shall examine*** the security architecture description to determine that it presents an analysis that adequately describes how the SFR-enforcing mechanisms cannot be bypassed.

538 Non-bypassability is a property that the security functionality of the TSF (as specified by the SFRs) is always invoked. For example, if access control to files is specified as a capability of the TSF via an SFR, there must be no interfaces through which files can be accessed without invoking the TSF's access control mechanism (such as an interface through which a raw disk access takes place).

539 Describing how the TSF mechanisms cannot be bypassed generally requires a systematic argument based on the TSF and the TSFIs. The description of how the TSF works (contained in the design decomposition evidence, such as the functional specification, TOE design documentation) - along with the information in the TSS - provides the background necessary for the evaluator to understand what resources are being protected and what security functions are being provided. The functional specification provides descriptions of the TSFIs through which the resources/functions are accessed.

E.g. IEC 62443-4-1 - Architecture/Process

Use of a MILS system provides partitions as natural application containers with "trust boundaries"

7.3 SR-2 –Threat model

7.3.1 Requirement

All products shall have an up-to-date threat model with the following characteristics:

a) correct flow of categorized information throughout the system;

x) trust boundaries;

y) processes;

z) data stores;

8.3 SD-2 – Defense in depth design

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.

MILS gives you at least one extra layer of defense

8.7 SD-6 – Secure design industry recommended practices

8.7.1 Requirement

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:

a) least privilege (granting only the privileges to users/software necessary to perform intended operations);

b) using proven secure components/designs where possible;

c) economy of mechanism (striving for simple designs);

d) using secure design patterns;

e) attack surface reduction;

f) all trust boundaries are documented as part of the design; and

g) removing debug ports, headers and traces from circuit boards used during development from production hardware or documenting their presence and the need to protect them from unauthorized access.

E.g., J3061 (input for ISO 21434)

System design view

8.4.3 Refine Functional Cybersecurity Concept into Technical Cybersecurity Concept

In the Concept phase, a Cybersecurity Concept was defined. In this task, that Cybersecurity Concept is analyzed, along with the System-level Vulnerability Analysis, to identify the System Functions that are at most risk relative to a potential Cybersecurity event. This analysis, and the determination of the high priority functions/data for Cybersecurity, will be used to create a Technical Cybersecurity Concept that defines specific technical decisions that will be made at the System level relative to a Cybersecurity design to protect these high-priority functions/data. Examples include:

Isolation = partitioning

- Isolation of specific functions For example, should a calculation for particular function be done on a separate circuit?
- Use of countermeasures (e.g., encryption, decryption).
- Not storing a copy of current ***GPS*** location on system.
- Defense-in-depth strategy, etc.

8.4.4 Specify Technical Cybersecurity Requirements

Once the Technical Cybersecurity Concept has been defined, the specific system requirements can be identified. In order to do this task, there should be a catalog of which specific functions (e.g., activation of airbags, braking, steering, etc.) will be performed by the system. In addition, a System Context is created to define the interfaces and functions within the system. These include:

- Hardware and software interfaces,
- Data flows,
- Data storage,
- Data processing,
- Functions which support Cybersecurity functionality.

The MILS OS helps to establish clear data flows and allocation of storage, cybersecurity functions

2. Protect:

[J3061 Appendix, p. 113]

Every component in a subsystem that impacts (i.e., initiates/terminates/provides inputs for) key Cybersecurity functions and stores/transports Cybersecurity critical data should be secured by a security mechanism that is appropriate for the level of risk.

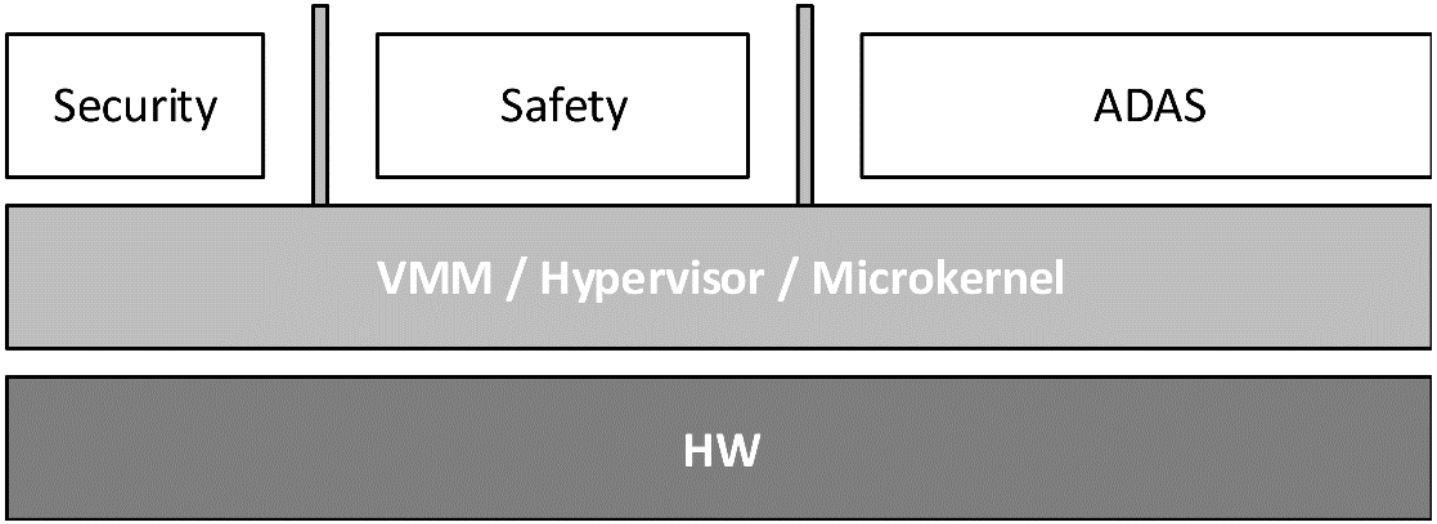
Security Mechanism - System functionality that preserves the desired Cybersecurity properties of data and routines. Security mechanisms include 1.) authentication mechanisms (is an agent allowed access to a certain resource) 2.) integrity mechanism (prevent unauthorized modification/writing of messages) 3.) confidentiality mechanism (preventing unauthorized reads of data).

Security Mechanisms:

A MILS design isolates these entry points of known attacks

a. Isolation/partitioning of systems that have external access (e.g., Wi Fi, Bluetooth, OBD) from safety-critical systems and systems that can have important impacts on the operation of the vehicle.

AUTOSAR adaptive



Security/safety architecture for mixed-critical system

AUTOSAR GbR, „AUTOSAR_EXP_SafetyOverview.pdf,“ 2018. [Online]. Available: https://www.autosar.org/fileadmin/Releases_TEMP/Adaptive_Platform_18-03/General.zip. Section 2.3.3.1

Architecture and composition in security standards

- Diverse security (and safety) standards recognize that it makes sense to have architectural design into components and their interactions
- Functional challenges:
  - Systems programming judgement
    - when to use address spaces or not, when to use hypervisors or not
- Certification judgement
  - What other certification schemes to (critically) trust
    - E.g. IEC 62443 / ISA referencing equivalent standards / Common Criteria
    - Timing of other certifications
  - How to define products and certification evidence whose assurance can be plugged into other systems

ISASecure SDLA

Secure Development Lifecycle Assurance, based on IEC 62443-4-1

SDLA-312 DSD: Detailed Software Design

	X	SDLA-MIV-5	COTS Operating Systems	If the product includes a Commercial off the Shelf (COTS) operating system, then the operating system shall either meet the requirements of this development phase or be certified to Common Criteria EAL 3 or higher or be certified to a comparable security standard, or compensating controls must be included in the product to ensure that security vulnerabilities in the operating system do not result in vulnerabilities above a certain severity level in the product.	

Checking that your MILS OS has / will have Common Criteria EAL3 or higher certification can future-proof your system (NB: required for SDLA levels 2-4).

Cybersecurity standards and certification - the challenges for architecture and composition

- EU has rich industry; including embedded systems
- When you produce a domain-independent component (e.g. OS), then you have to certify it according to domain-dependent standards
  - a domain-independent security standard for components could be a good thing
  - the EU could suggest to industry-specific standards to accept domain-independent security certification for components
- Technically, Common Criteria (CC) is an industry-neutral security standard
  - Common Criteria process can be time-consuming
  - We try to simplify the process by providing a protection profile for separation kernels
    - Join us at CC Users Forum, http://ccusersforum.org/ for the separation kernel technical community
      - This is the first community-based CC standardization effort for an OS component of an embedded system
    - Or join us at http://mils.community/

CERTMILS Contract No: 731456

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

If you need further information, please contact the coordinator:

Technikon Forschungs- und Planungsgesellschaft mbH
Burgplatz 3a, 9500 Villach, AUSTRIA
Tel: +43 4242 233 55 Fax: +43 4242 233 55 77
E-Mail: coordination@certmils.eu