Security Domain Separation as Prerequisite for Business Flexibility

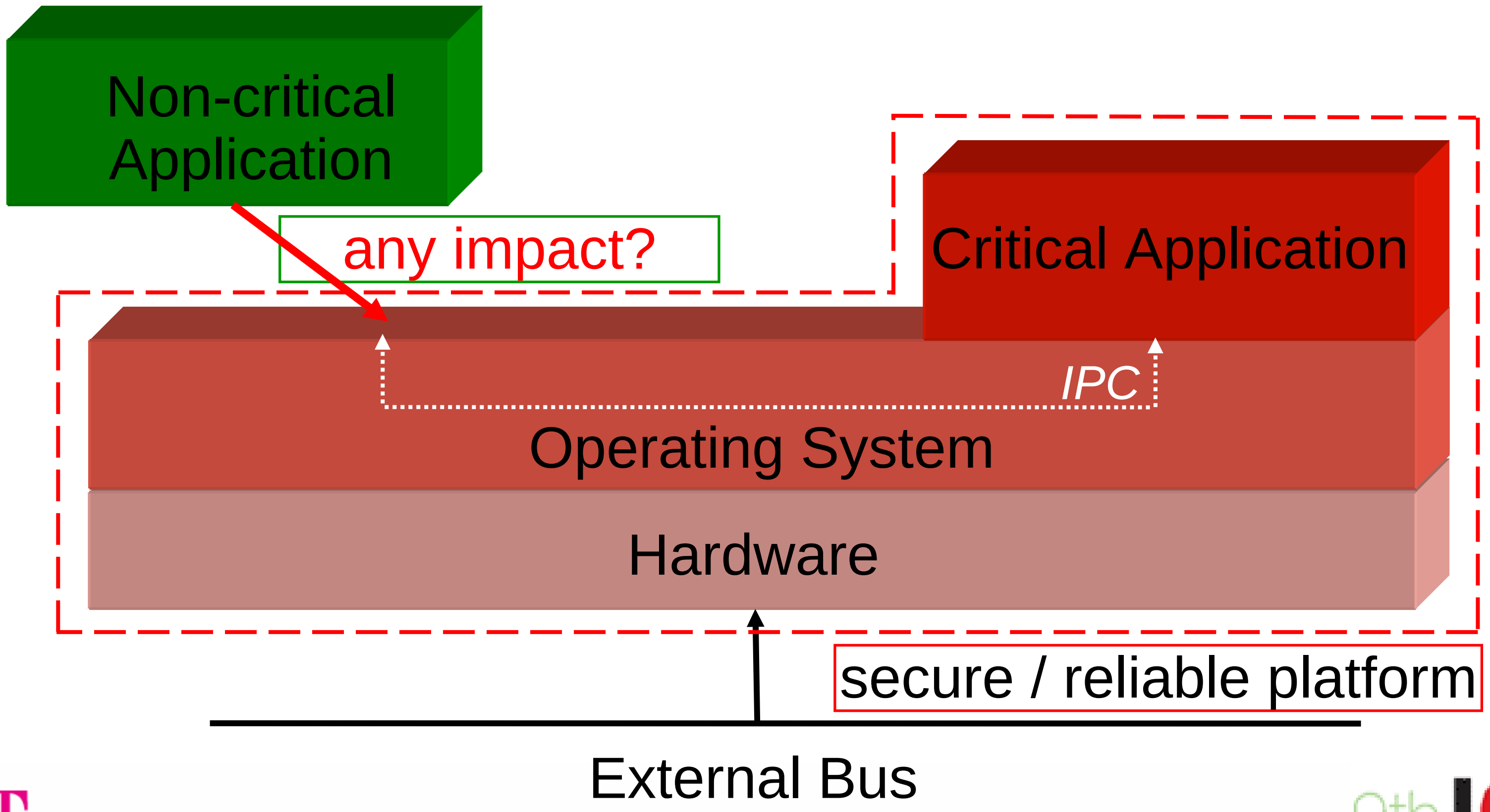
Igor Furgel
T-Systems

What are we speaking about?

- What is a Security Domain and what do we need it for?
- Impact on Security Target
- Means of Domain Separation
- Domain Separation and ADV_ARC
- Responsibility and Evidence: Developer vs. Operator
- Conclusion

What is a Security Domain?

- Let us imagine a classical situation:
  - there is a secure/reliable platform and
  - we need to add a non-critical application onto this platform



What is a Security Domain?

- What is common at such a constellation?

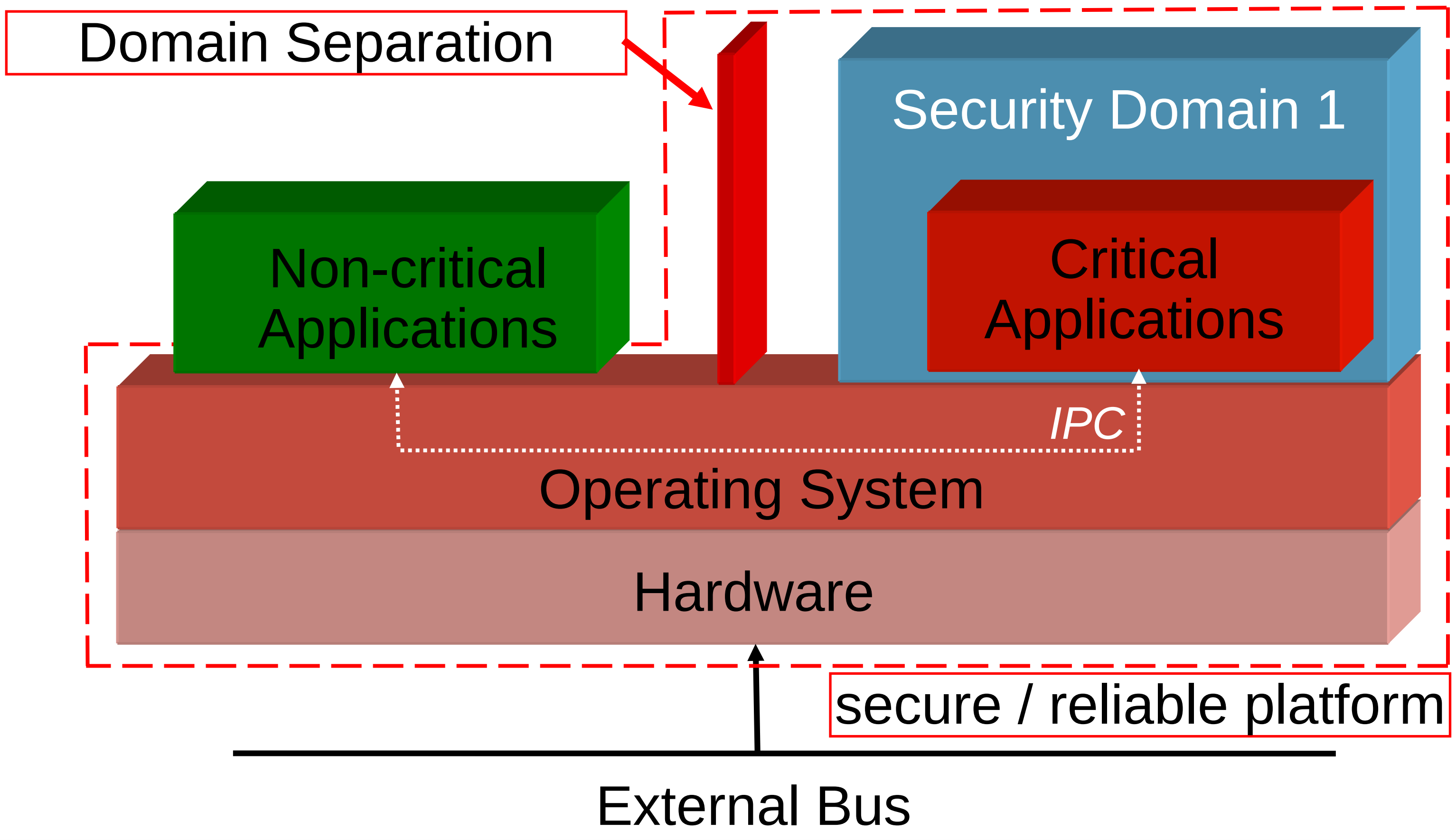
  => Common is <u>resources sharing</u>, when some applications shall or have to use <u>same</u> resources

- Resources sharing opens an opportunity of interacting between applications and, hence, their mutual impacting

- The key idea behind the property ‘Domain Separation’ is that the TSF creates Security Domains for use by potentially harmful entities and that these domains are kept separate from each other

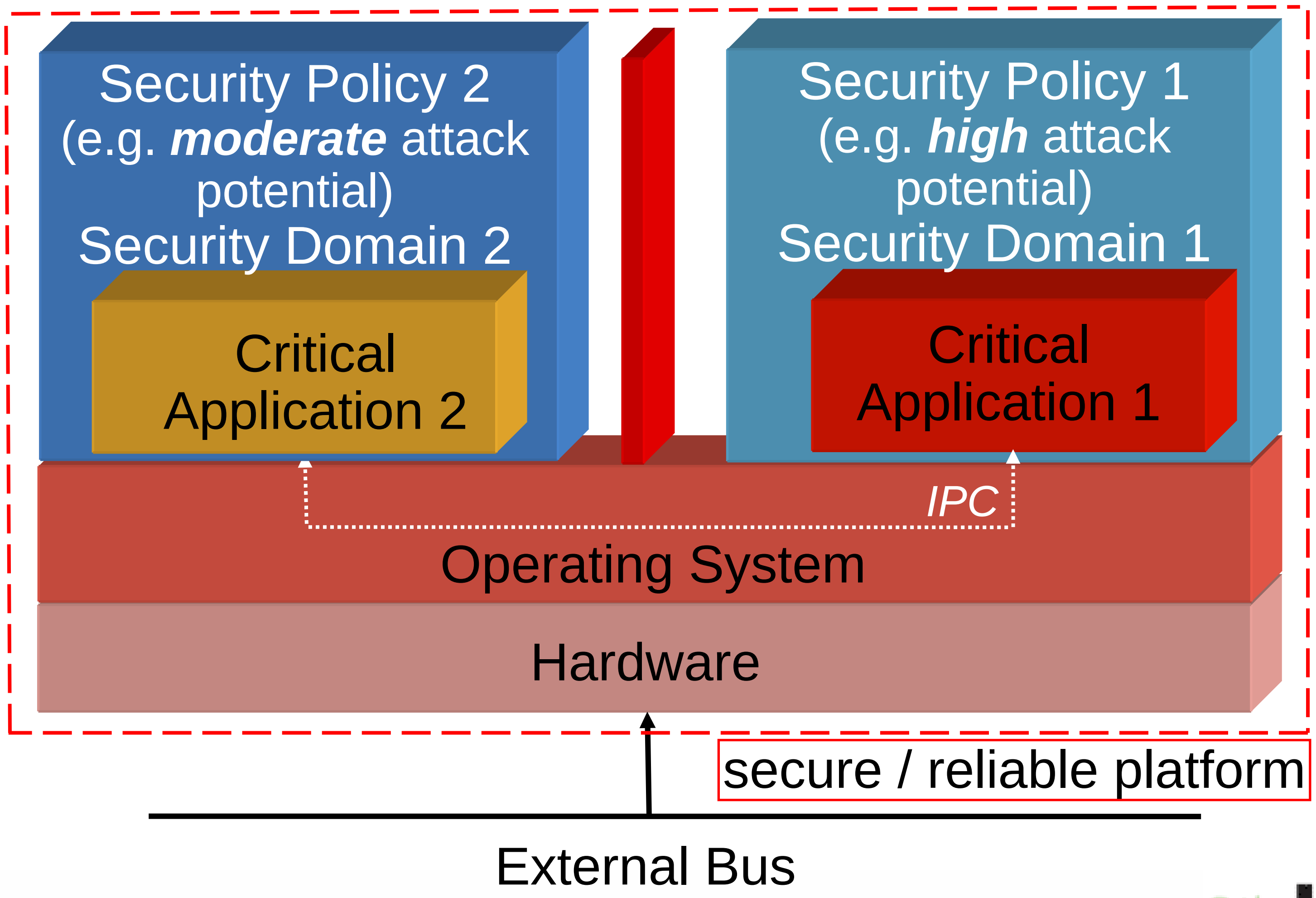
What is a Security Domain?

- Good example: field upgrades on mass market products



What is a Security Domain?

- Heterogeneous security policies enforced by same TOE



Security Policy 2 (e.g. *moderate* attack potential)
Security Domain 2
Critical Application 2
Security Policy 1 (e.g. *high* attack potential)
Security Domain 1
Critical Application 1
IPC
Operating System
Hardware
secure / reliable platform
External Bus

What is a Security Domain? Formal Definition

- A security domain is a confined active physical and/or logical unit where a single and homogeneous security policy is valid and applied. This security policy controls the behaviour of security services being provided in the context of this security domain.
- The main generic characteristics of a security domain are the following:
  - *a security domain as a whole represents an encapsulated unit and can be considered as an object;*
  - *internal and externally visible actions and reactions of this unit represent its well-defined properties;*
  - *communication between such objects occurs by well-defined (i.e. syntactic and semantic) messages and implements the relationships between the objects.*
- This definition of security domain covers the TSF self-protection as well as the relevant service (resources) offered to other entities.

Impact on Security Target

- Speaking the Common Criteria language, a security policy for a TOE is defined by a set of security functional and assurance requirements {SFRs + SARs} chosen for this TOE within the Security Target.
- Hence, a ST defines a <u>homogeneous</u> security policy and, therefore, effectively exactly <u>one security domain within the TOE</u>.

- Even in this simple case, there are indeed two security domains (from the TOE's point of view):
  - the first one is defined within the TOE by the ST: {SFRs + SARs},
  - the second, *implicit* one is the remaining world outside the TOE: this default security domain always exists and shall enforce the security policy as expected by the environmental security objectives stated in the ST.
- The stripline is the physical/logical scope of the TOE. The TSF shall maintain this 'shell' (=> ADV_ARC).

Impact on Security Target

- Let us imagine that
  - The values of assets secured by a product are different or
  - A product is physically distributed and its single parts are operated within differently restricted environments (e.g. in open field and inside a company).
- In such a common case there are two opportunities to define a security policy for this product:
  - either a homogeneous security policy <u>at the most restrictive level</u> (this is the approach of the current CC)
  - or a <u>heterogeneous</u> security policy <u>fittingly to the situation</u>.
- For a <u>heterogeneous</u> security policy we need <u>different sets</u> of {SFRs + SARs} within a product and, hence, different security domains inside same TOE.
- For such a case, an effective, TOE-internal separation of these security domains becomes a necessity.

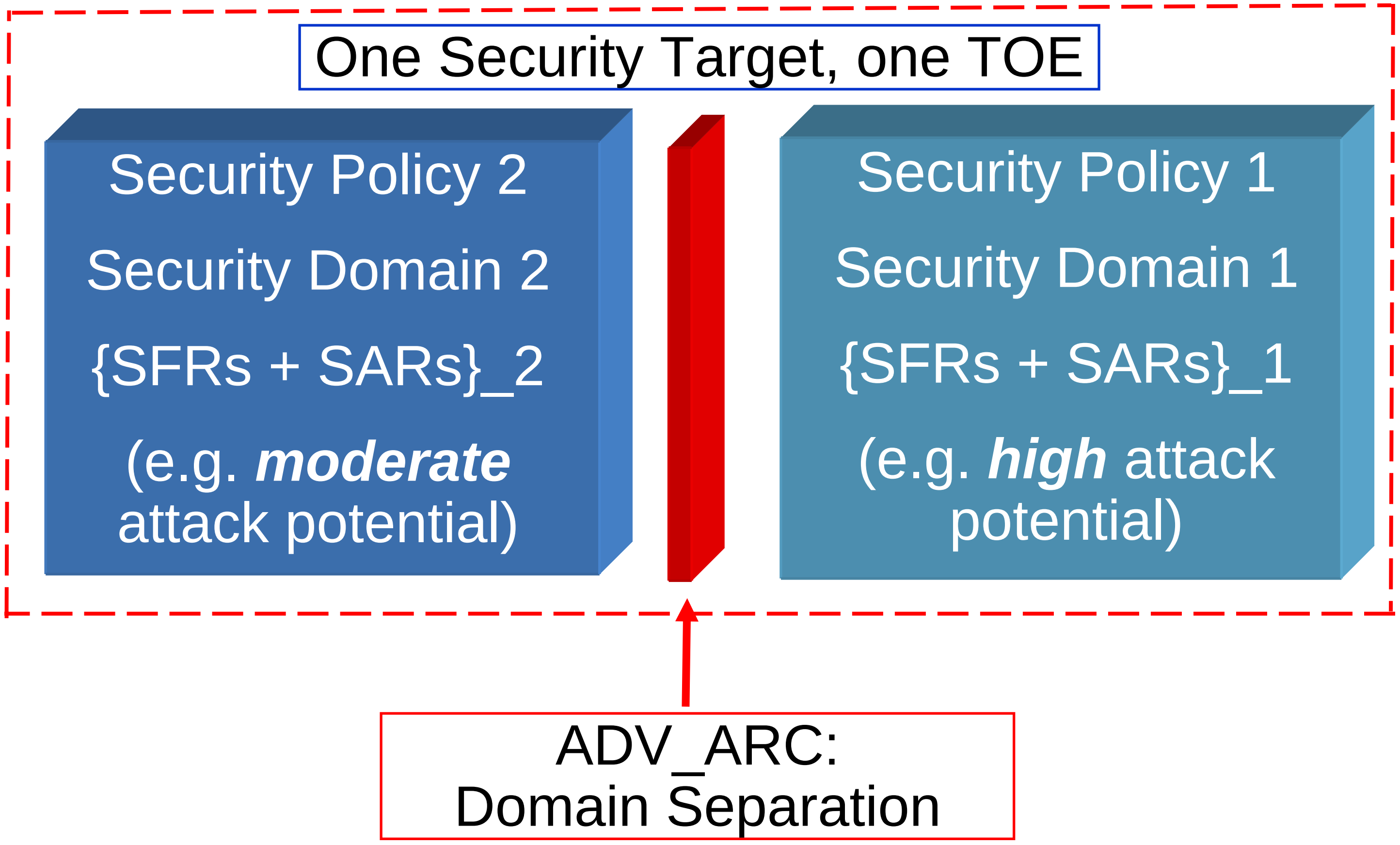
Impact on Security Target
Suggestion for the next CC version

- A Security Target shall permit definition of different security domains for same TOE

- A dedicated set of {SFRs + SARs} (and, if appropriate, parts of SPD and Security Objectives) shall be attributed to each security domain within the ST (of course, there could be overlapping and hierarchical requirements)

- If a ST does define different security domains, the family ADV_ARC must be part of the assurance package chosen.

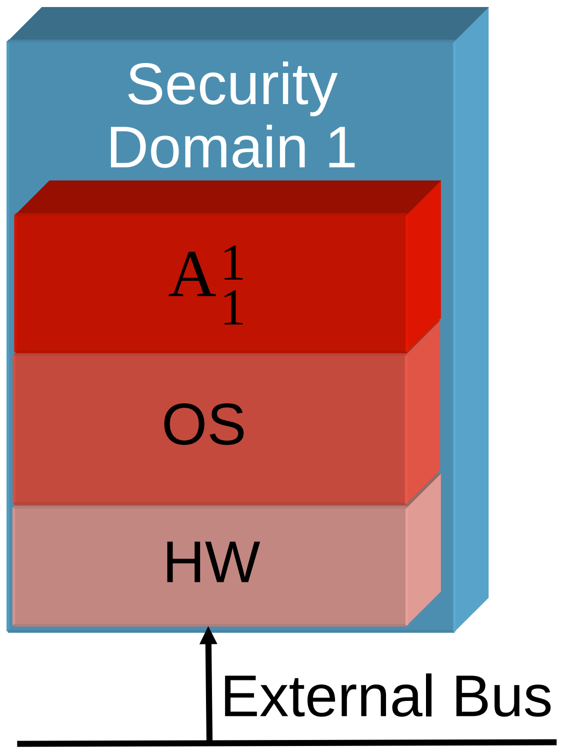
Impact on Security Target
Suggestion for the next CC version



Means of Domain Separation

- Domain Separation can be achieved on physical and logical ways
  - In case of physical separation, all communication passes through the external bus and the domain separation is trivially defined (but not trivially enforced).
  - In case of logical separation, this requires the operating system to be of a different nature than the applications running on it: The OS itself shall provide special services enforcing a Domain Separation.

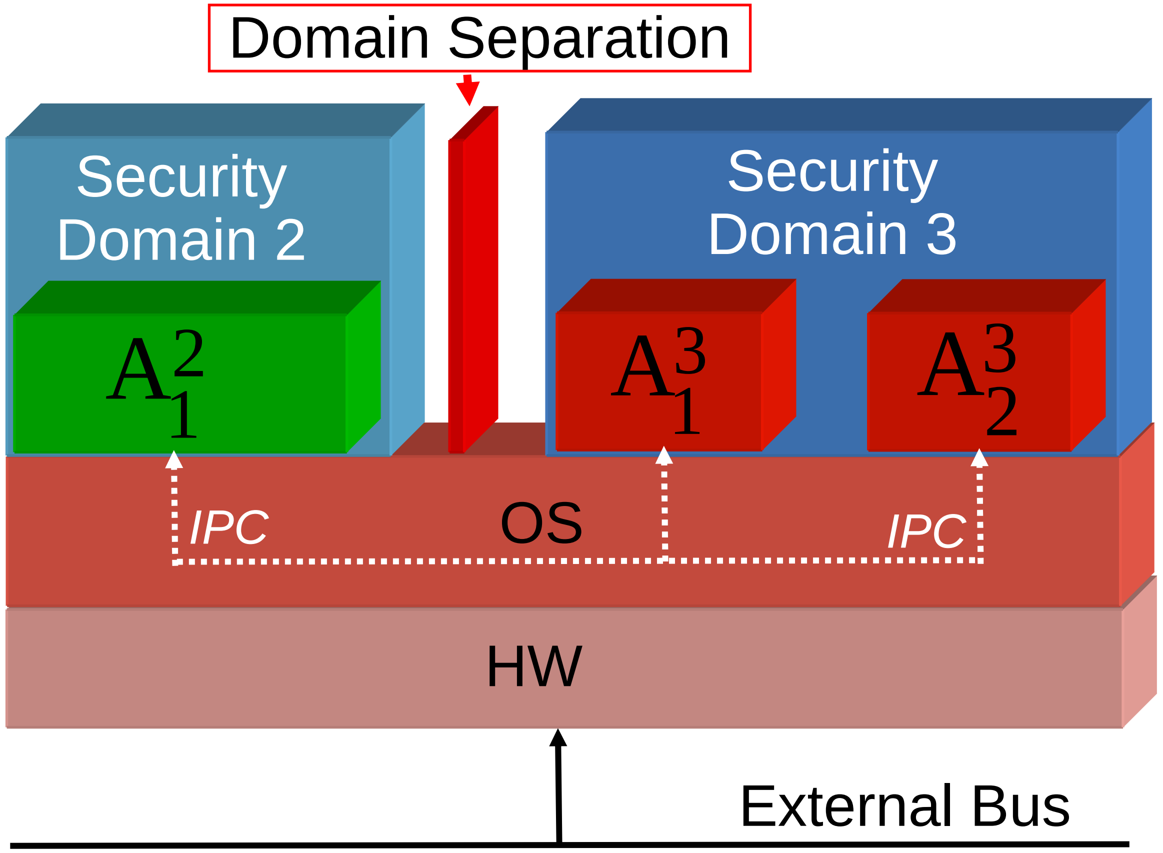
Means of Domain Separation: Physical Separation



- Security Domain 1
- Application A^1_1 (single module)
- Operating system (OS) maintains resources
- Hardware (HW) provides physical protection and <u>may support</u> OS (e.g. NMI)

- Segregating modules:
  - Keeping security through well-defined interfaces (*external* separ.)

Means of Domain Separation: Logical Separation



- Security Domain 2
  - Application A^2_1 does not trust $A^3_1 + A^3_2$
- Security Domain 3
  - Applications $A^3_1 + A^3_2$ do not trust A^2_1
- Operating system manages all resources incl. IPC
- Hardware provides physical protection and must support OS (MMU)

- Segregating modules:
  - Maintaining security through well-defined interfaces (*external* separ.)
  - Divide-et-Impera state-space of complex applications (*internal* domain separation)

Means of Domain Separation

- Technical means may be used for Domain Separation:
  - Access Control on the domain resources
  - Information Flow Control on inter-domain communication
  - Temporarily confined use of shared resources (‘object reuse’)
  - Physical protection

- Organisational prerequisites being helpful for Domain Separation:
  - Comprehensive architectural concept
  - Controlled usage of well-defined design and programming rules
  - ‘Object-oriented’ hard- and software design
  - Well-defined inter-domain communication format(s)

Security Domain Separation and ADV_ARC

- An effective security domain separation is one of the crucial architectural decisions.

- Firstly, it encompasses other architectural security properties like
  - Self-Protection being one of the *means* for Domain Separation, and
  - Non-Bypassing being one of the *effects* of Domain Separation

- Secondly, a domain separation decision may significantly impact the general security design concept of the TOE by
  - impacting the entire life cycle of and, hence, the business model for the product (e.g. SW upgrades),
  - influencing the design specification (ADV_TDS), and
  - representing an important input for vulnerability analysis (AVA_VAN)

Security Domain Separation and ADV_ARC

- So, it is evident that an evaluator needs a description
  how the domain separation is achieved (separation mechanisms)
  in order to understand efficiency of the separation and to become able to perform vulnerability analysis.

- The current assurance component ADV_ARC.1 finally requires the evaluator to determine that the developer's description of the security domains takes into account all of the SFRs claimed by the TOE (cf. CEM)
- There is no formal requirement to analyse how the domain separation is achieved (merely a hint in Application Notes in CEM).

Security Domain Separation and ADV_ARC Suggestion for the next CC revision/version

- ADV_ARC.1.2C (Part 3): The security architecture description shall describe the security domains maintained by the TSF consistently with the SFRs and how the domain separation is achieved.
- ADV_ARC.1-2 (CEM): The evaluator ***shall examine*** the security architecture description to determine that it describes the security domains maintained by the TSF and how the domain separation is achieved.

- Since
  - Self-Protection is one of the *means* for Domain Separation, and
  - Non-Bypassing is one of the *effects* of Domain Separation,

  we also suggest including the content of ADV_ARC.1.4C and ADV_ARC.1.5C in ADV_ARC.1.2C.

Responsibility and Evidence: TOE Developer vs. TOE Operator

- Who is responsible for Domain Separation and who brings evidence for its security properties/reliability?

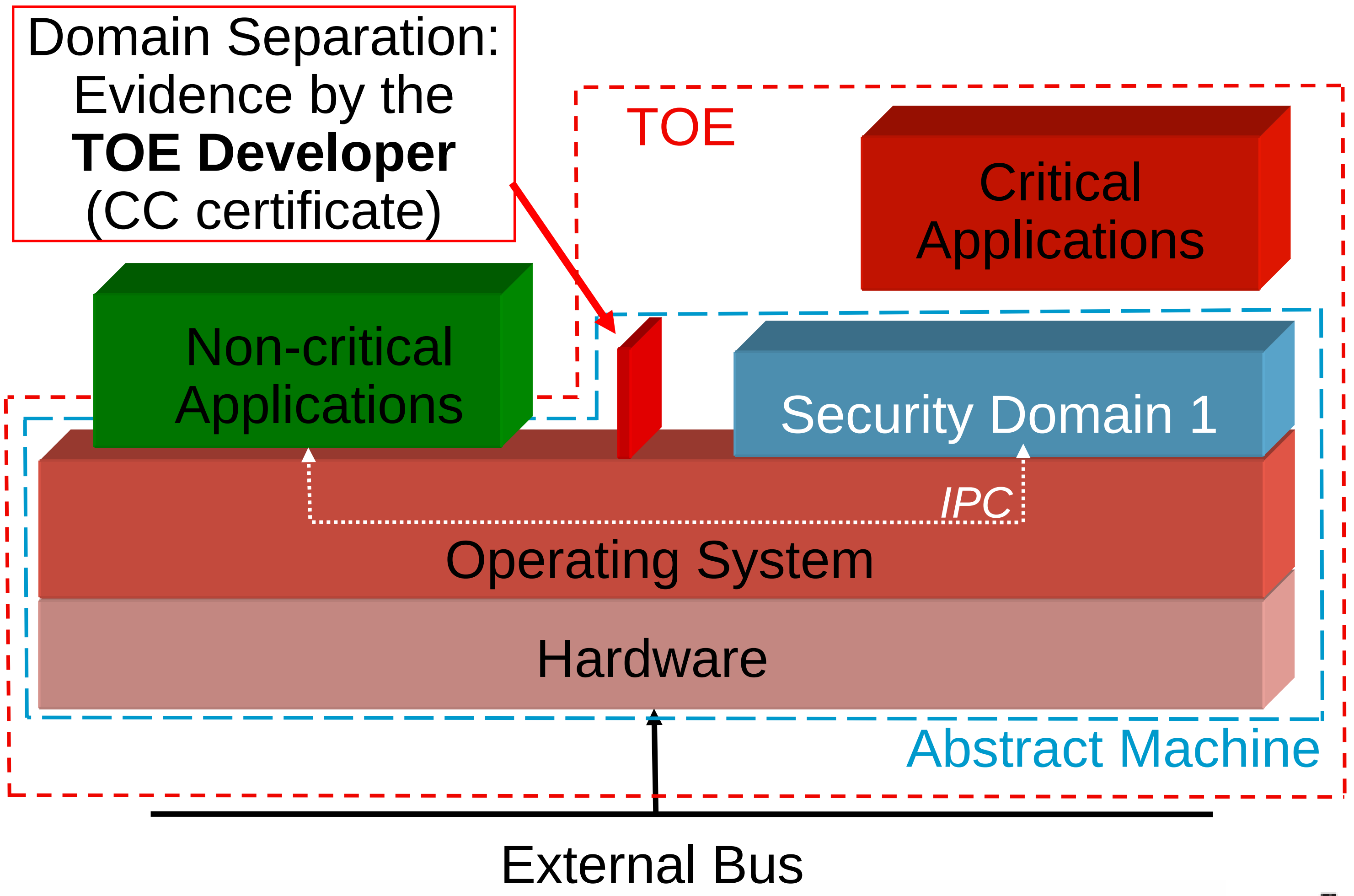
- The TOE Developer implements an Abstract Machine managing
  - resources to be shared among applications and
  - communication between them.

- This Abstract Machine shall enforce domain separation;
  an application is usually not able to manage all these resources by its own means.

Responsibility and Evidence: TOE Developer vs. TOE Operator

- If the relevant separation mechanisms are in the scope of a security evaluation, the evaluation and certification bring the evidence for the domain separation.
- I.e. the TOE Developer brings this evidence in this case.
- The Security Domain Provider (installing and configuring the Security Domain) and the TOE Operator shall merely follow the respective instructions of the TOE Developer.

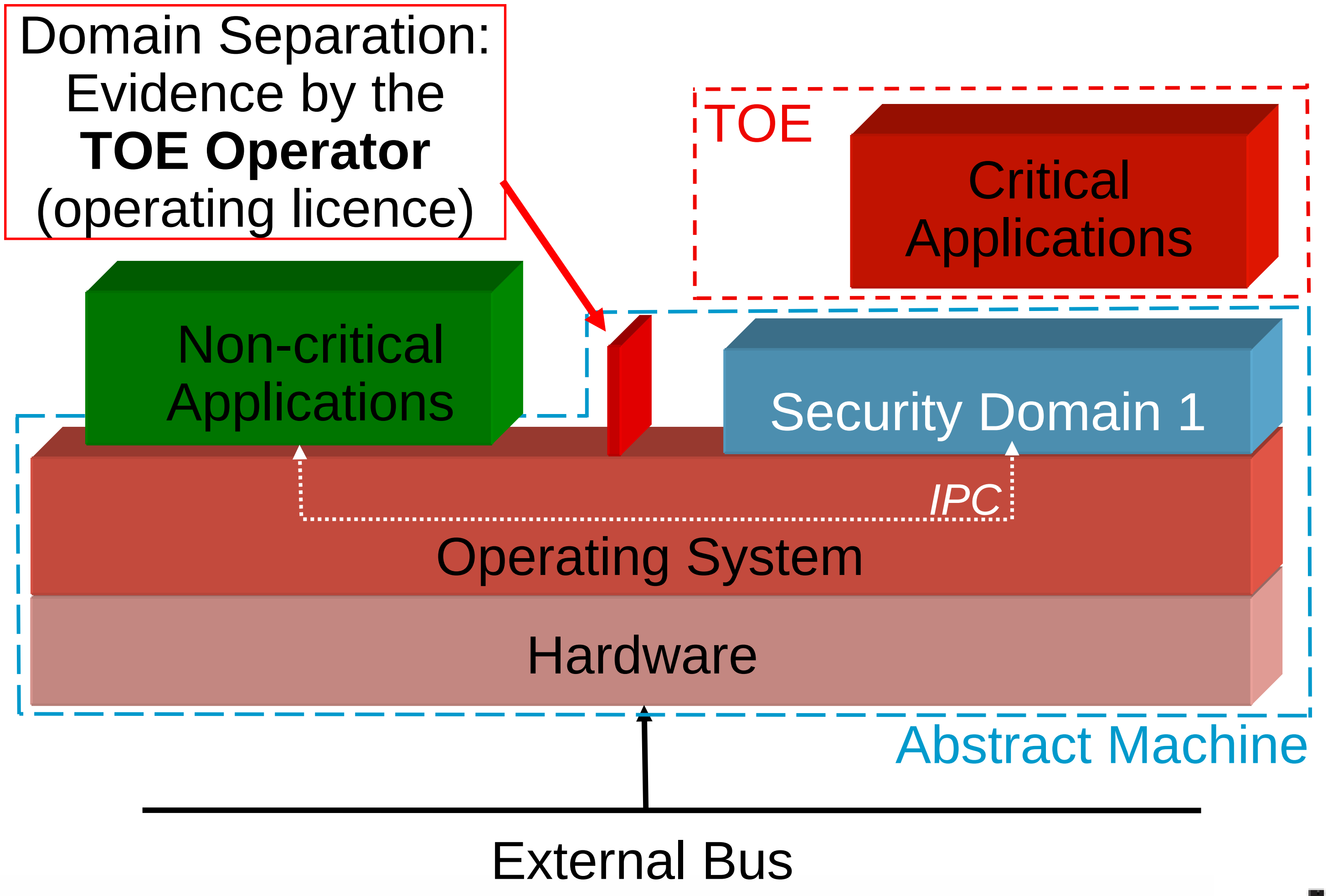
Responsibility and Evidence: TOE Developer vs. TOE Operator



Responsibility and Evidence: TOE Developer vs. TOE Operator

- If the relevant separation mechanisms are assumed being provided by the TOE's IT environment, the entire responsibility lies on the TOE Operator
- He has to operate an Abstract Machine (TOE's IT environment) evidently separating Security Domains on a sufficient security level (i.e. according to the security policy for the entire product/system which also might include an AVA_VAN assurance requirement).
- But how can the TOE Operator get such an evidence for the TOE's IT environment?
  Again, rather by a security certification of the related domain separation mechanisms (the context: operating licence)!
- There is no way to circumvent an appropriate certification of the separation mechanisms. The only question is who provides the evidence: the TOE Developer or the TOE Operator!

Responsibility and Evidence: TOE Developer vs. TOE Operator



Domain Separation and Business Flexibility

- Domain Separation facilitates business and service flexibility
    - by ‘free-hand’ adding additional and
    - modifying/deleting existing applications

  on a running trusted platform
    - **without** security re-certification and
    - **without** recall of the product(s).
  - These modifications can be done **directly in the field** by the product operator or **in a trusted environment** by the service provider (depends on business model)

- Opportunity for heterogeneous Security Policies within same ST
  - This approach enables defining merely **one ST** for a product where different security needs really exist (e.g. ***medium* and *high*** resistant domains **within same TOE**).
  - It also enables issuing merely **one security certificate** for this TOE.

- Price to be paid
  - Additional implementation of separation mechanisms or/and special architecture (e.g. physical separation) as well as
  - Additional evidence for effectiveness of the domain separation

The benefits are worth the price!

Thank you for your attention!

Dr. Igor Furgel

T-Systems GEI GmbH
ICT Security

Rabinstrasse 8
53111 Bonn

☎ +49 (228) 98410
💻 igor.furgel@t-systems.com