INVESTIGATION INTO A LAYERED APPROACH TO ARCHITECTING SECURITY-INFORMED SAFETY CASES

Kateryna Netkachova, City University London and Adelard LLP, London, UK

Kevin Müller, Airbus Group Innovations, Munich, Germany

Michael Paulitsch, Thales Austria GmbH, Vienna, Austria

Robin Bloomfield, City University London and Adelard LLP, UK

Abstract

The paper describes a layered approach to analysing safety and security in a structured way and creating a security-informed safety case. The approach is applied to a case study – a Security Gateway controlling data flow between two different security domains implemented with a separation kernel based operating system in an avionics environment. We discuss some findings from the case study, show how the approach identifies and ameliorates important interactions between safety and security and supports the development of complex assurance case structures.

1. Introduction

Assurance Cases in their many forms (safety, security, dependability and reliability cases) have been around for many years with considerable work done around structuring them in a variety of ways [1][2][3][4][5]. In this study, we are developing a practical approach to structuring cases based on the idea of compositionality and layered assurance [6][7]. This is particularly useful when dealing with complex security-informed safety cases, which provide justification of safety taking into full consideration the impact of security.

A detailed safety analysis and the production of a safety case are required by various standards in automotive [8], railway [9], nuclear [10] and other industries. Most of these standards focus solely on safety without taking security into account. However, there is a growing realization that security considerations can have a significant impact on safety especially with the increasing openness of systems. For many cyber-physical systems, analysis of safety and security is combined to achieve high assurance. Our experience and previous research [11] has shown that a significant portion of a safety case is affected by considering security. Introducing security aspects increases the complexity of the resulting security-informed safety case and one of the possible structuring mechanisms for dealing with that complexity is a layered approach to constructing cases that we are developing.

This paper provides an overview of our approach illustrating it on an example from the avionics domain, where the combined security-informed safety assessment is particularly important. Aviation has been implementing systems by following the safety standards [12][13] and recently by using the safety-driven design-approach of Integrated Modular Avionics (IMA) [14]. IMA relies on the concept of partitioning between applications. Similarly, avionic industry has recently issued security standards [15][16]. In terms of the system design, the security-driven design approach of Multiple Independent Level of Security (MILS) is used with partitioning and a separation kernel to control information flow [17]. Building on the safety case approach and taking into account security considerations, in this study we are creating an integrated security-informed safety case for a MILS-based gateway controlling information flow between aircraft security domains.

The paper is structured in the following way. The overview of the layered approach to architecting security-informed safety cases that we are developing is provided in Section 2. A brief description of the Security Gateway use case is provided and the application of the layered approach to the case study illustrated in Section 3. Conclusions, some finding from the case study and next steps for future research are outlined in Section 4.

2. Security-informed Safety Case

2.1 General Concept

The approach we are developing is based on the use of structured assurance cases for communicating and building confidence in the safety and security properties of the system. Structured assurance cases are used in a wide range of industrial domains. Our current method is based on a concept of Claims, Arguments and Evidence (CAE) [1], which can be related to the approach developed by Toulmin [2].

Security considerations have a significant impact on various aspects of safety justification. It is necessary to identify security properties as well as safety properties, to demonstrate compliance with both security and safety standards, and to consider a broader set of potential hazards, threats and vulnerabilities.

Our previous research [11] has shown that a significant portion of a security-informed safety case will need to address security explicitly. In some instances this will lead to substantial changes to the system design, the implementation process and the justification. For example, the following areas are particularly significant from a security perspective and are evident in large scale systems, such as an aircraft:

- Supply chain integrity
- Confidentiality of the process and product
- Combination of specific aspects of addressing safety and security design approaches and life cycle processes of certification and updates, such as stable and infrequently changing system design and certification approaches addressing safety combined with requirement of frequent security subsystem updates.
- Issues of lifecycle threats and malicious threats to evidence, e.g.:
    - Malicious events after system deployment that will change in nature and scope as the threat environment changes.
    - Weakening of security controls as the capability of the attacker and technology changes. This may have major impact on proposed lifetime of installed equipment and design for refurbishment and change.
- Design changes to address new user interactions, training, configuration, and vulnerabilities. This might lead to additional functional requirements that implement security controls.
- Possible exploitation of the device/service to attack itself or others.

In order to address these additional security risks for an avionics example, it is necessary to combine safety and security risk assessment. Because many systems already have safety justifications with corresponding risk assessments we are developing an adapted process to make them security-informed, as opposed to the common approach of starting with security issues and then merging them with safety standards. As a result, the presented approach brings the security-informed safety case perspective into the avionics domain.

2.2 Security-Informed Safety Case Architecture

The justification of a security-informed safety case can be complex, or at least complicated, as it combines the claims from adaptation, supply chain deployment, and hazard and vulnerability analysis. As one role of the case is to communicate effectively, one needs to balance both the risk of abstracting away important details and the risk of the important details being lost in a sea of other details.

The works of Rushby and Delong [6][7]raise the idea of compositionality and layered assurance. The goals of the approach are manifest in the LAW series of workshop between 2005 and 2012 [18]. These explore the “bold proposition that it is possible to build assured systems from compositions of previously assured components, while being able to derive the system level properties (e.g., safety & security) systematically from the properties of the components”. LAW spans the theoretical, engineering, and certification challenges to be met in making compositional assurance for such systems a reality. They use the term "layered" assurance to encompass diverse manifestations of combined assurance, including composition (of assured components), incremental certification (incremental cost for incremental change), abstraction layers (building upon assurance of lower layers), and

polymorphism (common assurance of variants, such as among members of a product family). MILS is one approach to achieving the goals of compositional assurance.

Abstraction is one of the key structuring mechanisms and we have experimented with various levels of abstraction when creating security-informed safety cases. We call those levels layers of assurance, because within each abstraction level the assurance is provided. We identified the following main layers of assurance:

- L0 Policy and requirements – the highest level of abstraction where the system represents its requirements, and defines safety and security policies and their interaction;
- L1 Architectural layer – the intermediate level where the abstract system components and architecture are analysed;
- L2 Implementation layer – the detailed level where the implementation of specific components and their integration within the specific system architecture are scrutinised.

These layers of assurance fit well the layered system design approach of aerospace described in ARP 4754 [13] and ED-203 [16] combined with the compositional approach of MILS [17] and IMA [19].

The following chapter instantiates and discusses each abstraction layer in application to the MILS gateway use case, a potential subsystem in future aircraft system architectures.

3. Analysis of a Security Gateway

3.1 Overview of the Gateway Use Case

As a use case for our study, we use a gateway function. Usually security gateways connect two or more security domains [20] to each other and control the information flow according to a given information flow policy. It is important to note that controlling information flow in the context of an airplane focuses more on ensuring the integrity and availability of one domain (by maintaining the integrity of safety-assured application and as a consequence safety) rather than protecting the confidentiality of the data propagating from one domain into another. The careful joint consideration of the security and safety needs and properties of this gateway is the basis for our security-informed safety analysis.

Our gateway is implemented by using the design concept of Multiple Independent Layers of Security (MILS). MILS bases on the properties of separation and controlled information flow. In current MILS proposals these two properties are achieved by a special operation system layers. This operating system provides separation by the concept of partitioning. Partitions are isolated runtime environments for applications. Since a meaningful MILS system only works with allowed interactions among application, the operating system also allows a controlled information flow between partitions. Applications can implement further layers of security, e.g. cryptographic algorithms or more sophisticated data processing, or entire other function such as image processing.

The gateway is intended to filter application-level data traffic for the TFTP and HTTP protocol. It achieves its data filtering by a cooperating of several partition applications allowing to interact in a certain way with each other. Figure 1 shows the abstract system design [21]. Each solid box represents one partition provided by the Separation Kernel. In the use case implementation we use PikeOS [22], however the analysis remained generic as long as the Separation Kernel supports spatial and time partitioning. PikeOS is currently under Common Criteria evaluation targeting the Evaluation Assurance Level 5+ (AVA_VAN.5) [23], and hence, being a good foundation for secure developments. The arrows define the directed and controlled information flow among partition in order to allow interaction. Using the MILS architecture allows the definition and implementation of a local security and information flow policy for each partition. For example, each Receiver Component is in charge to analyse ingress traffic and forward it either to the TFTP or to the HTTP filter chain. Each filter chain's policy again is to filter these data packets according to the prevalent application-level protocol.

The security policy of the Audit partition is to gather audit records generated by other partitions and to store them securely in order to provide traceability.

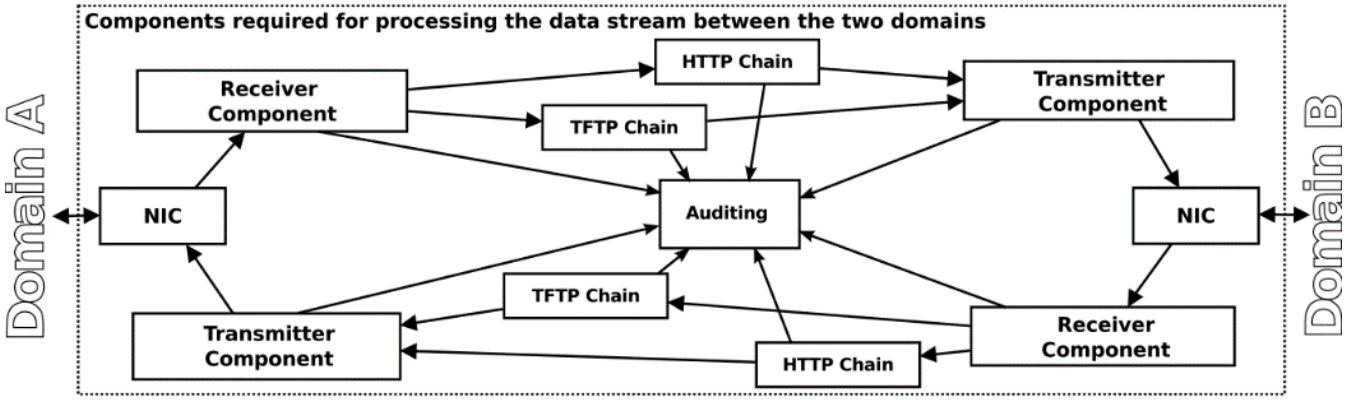

Figure 1. High-level View of Gateway Component

3.2 Application of the Layered Assurance Approach

Scope of the System

The scope of the security and safety analysis, of our use case is broader than just the gateway. We are interested in a system consisting of applications in two domains having different security levels, i.e. processing data of different classification. Each application belongs to a single security domain and can only communicate with applications from the other security domain via the security gateway.

Figure 2 shows a context data flow diagram representing the high view of such broader system. The double circle at the centre represents the gateway process that performs various operations (receives data streams, applies security policies, transmits the filtered content etc.).

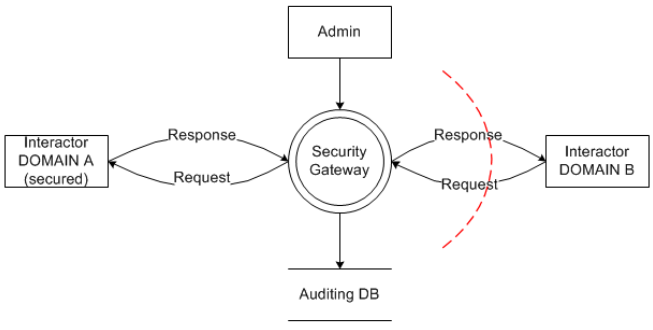

Figure 2. High-level data flow diagram of the system

Rectangles identify external entities that interact with the gateway. They comprise applications from two domains (A and B) and the administrators that can maintain the gateway while being in maintenance mode (a special operational mode). An open-ended rectangle indicates a data store where the logging and alerting data are stored by the gateway for later use.

Discussion of L0 on the Gateway

The first and most abstract level concerns requirements, policies and principles of the system, with the focus on the system safety, system security and their interaction.

The top-level claim involves introducing a security policy, considering a set of applications at different security classification, and safety criticalities associated with them. At L0 the abstract gateway enforces a security policy that puts constraints on inter-domain information flows. We need to undertake an analysis to show that the interaction and trade-offs are satisfactory.

It is unlikely that under all runtime circumstances one simple and static policy will be valid. There will be times of initialisation, special operational modes or changing threat levels that will impact the policy. For example at high levels of security threat the system might be adjusted to isolate high-safety applications from all applications of other domains. Alternatively, in times of operationally challenging conditions safety consideration could require an adaption of the security policies in order to allow manually transmitted messages by trusted

external entities to provide guidance and recovery strategies to the pilots.

Figure 3 provides an illustration of some of these considerations by showing different policy zones. At the bottom left we have an area of maximum operational benefit. The other areas indicate how certain threat level security concerns dominating e.g. the need to restrict the flows. In this case the safety analysis must show that these are acceptably safe even if they do cause higher workload or operational complexities. There is a corresponding zone where safety issues dominate and the security policy is the same or weakened. In this case, the security analysis must show that identified security threats are countered by other environmental properties during this time.

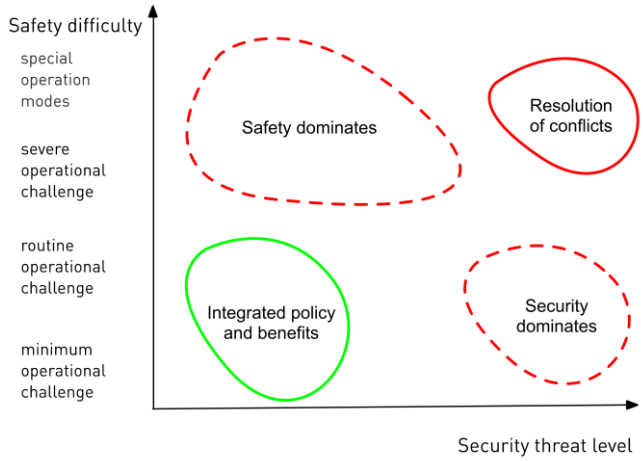

Figure 3. Defining integrated policy

Finally, the top right hand corner is a very uncertain and undecidable area where some special capabilities might be needed, e.g. in the form of a manual override to security policy enabling flows as well as a manual closing down on all non-essential information flows if threats were high and compromise was likely. Again, the consequences of any trade-offs need to be assessed during the analysis. In summary as we create the L0 case for the system we need to address the:

- modes of safety application
- operational modes of the gateway
- Impact of different threat levels
- attributes for the gateway's policy

At this level we develop a substance to the analysis of the policy interactions, we have an updated security policy and safety requirements as well as initial results from the risk analysis. Also, we identify some more details about modes of operation of the gateway and the overall system as well as availability and other attributes for the properties. The case structure created for the security gateway at L0 level of analysis is presented in Figure 4.

Discussion of L1 on the Gateway

At this level we analyse the components and the architecture of the system, which play important roles in achieving system objectives and enforcing the critical properties of the system.

To address security considerations, at this level of analysis we applied various methods of security analysis including:

- A security-informed guideword-based approach derived from the safety Hazop (Hazard and Operability) analysis;
- An analysis of trust relationships;

STRIDE (Spoofing, Tampering, Repudiation, Information Disclose, Denial of Service, Elevation of Privilege) [24], the Microsoft threat modelling approach.

Fragments of the analyses are presented in Tables 1, 2 and 3 below.

Table 1 provides an example of applying a security-informed Hazop guideword-based approach.

Table 2 presents a part of trust relationships analysis performed using the high-level system view (Figure 2). STRIDE methodology required us to analyse the system with respect to spoofing, tempering, repudiation, information disclosure, denial of service, elevation of privilege. A brief description of these scenarios is presented as part of the STRIDE analysis in Table 2.

Table 3 presents a part of trust relationships analysis performed using the high-level system view (Figure 2).

.

Figure 4. L0-level case fragment for the gateway in CAE notation

Table 1. Example of security-informed Hazop analysis for the gateway

N	Element	Guide word	Deviation	Possible causes	Consequences
Function considered: Connecting to the gateway					
1	Connect to channels	Other than	Application connects to a channel other than gateway‘s	Another application from the same domain pretends to act like a gateway	Man-in-the-middle attacks
2	Connect to channels	More	Too many messages are sent to gateway channels	Faulty or compromised application is sending too many requests	Denial of service
Function considered: Gateway filtering					
12	Filter messaged going through gateway	As well as	Additional messages are allowed to pass through the gateway	Error in filter specification or implementation	Leakage of confidential data Pass of malformed data (integrity)

Table 2. Trust relationship analysis of the gateway use case

Breach of Trust	Consequences	Mitigation
Gateway-Administrator	High Denial of service, loss of data integrity and confidentiality, man in the middle attack.	All security policies have to be operating and have to be identified by some authority. The gateway will only accept these security policies.
Gateway-Auditing	Medium Loss of accountability and nonrepudiation, possible impact on confidentiality of recorded data	Applications located in the domains can have their own logs documenting what they sent. No confidential data or data that can help facilitate an attack should be stored in logs.

Table 3. Fragment of the STRIDE analysis of the gateway

Threat type	Security property	Brief explanation	Use case examples
Spoofing	Authenticity	Impersonating someone else	1) Application from domain B pretends to be a gateway or an application from domain A and sends something to domain B users; 2) One application from domain B pretends to be another application from domain B and requests something from the domain A or gateway
Tampering	Integrity	Modifying data	One application from domain B intercepts and modifies the data send to or from another application
Denial of service	Availability	Denying or degrading service to valid users	Application from domain B sends too many messages to the gateway

Some of the attack scenarios created at this level of abstraction are illustrated in Figure 5.

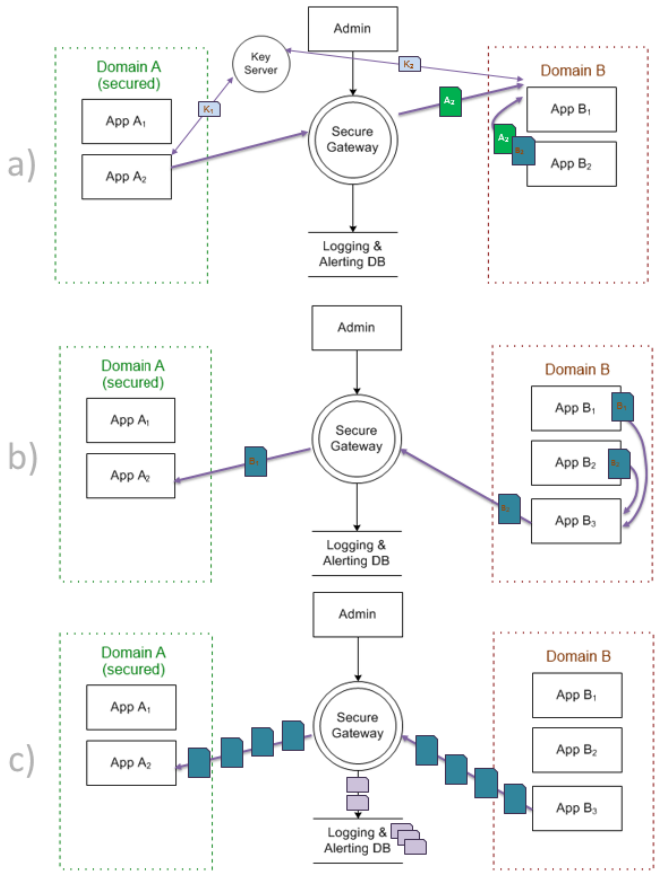

Figure 5. Attack scenarios: a) spoofing; b) spoofing and tempering; c) denial of service

The analysis conducted at the L1 level of abstraction showed that many critical aspects of the system are enforced by the separation kernel. Other important components of the architecture include the gateway application software, system monitor & audit component, system integrator, etc. The resulting L1 case is shown in Figure 6.

The L1 case considers the critical properties, related to the system functionality as well as other aspects (reliability, availability, etc.) and properties identified at L0. Each of the architectural components enforcing the critical properties of the system should be expanded further to demonstrate that the implemented subcomponents really enforce the corresponding properties. This type of analysis is to be performed at the next L2 level of abstraction.

Discussion of L2 on the Gateway

L2 represents the detailed implementation level. At this level we introduce all the technical information available about the actual system implementation.

To illustrate the approach let us consider one of the generic requirements that would be identified at L0 as part when analysing security policies of the system: all communication between domains of different security levels must be controlled in accordance with a system-wide security policy. Then, at the L1 level this is expanded into subclaims related to system components and architecture. One of such subclaims defined at L1 is: “All communication between domains is via the gateway” (Figure 6). The L1 analysis also shows that this subclaim is enforced by the defined information flow between partitions configured and assured by the PikeOS component within the system architecture. Therefore, the implementation details of both the gateway and the PikeOS need to be thoroughly analysed in order to demonstrate that the critical property is really enforced. This analysis is performed at the L2 level of abstraction.

In terms of implementation, the security gateway is composed out of user applications hosted by PikeOS. The gateway’s purpose is to control all information flow between applications located in different security domains according to the system-wide policy. All applications are supposed to communicate in accordance to the settings specified in the PikeOS system configuration file (VMIT file). This file is configured in an XML format by the system integrator and converted into a binary form. Then all software application binaries (with binaries of the security gateway component being part of them), PikeOS binary objects (microkernel, platform support package, PikeOS system software) and the PikeOS system configuration file are assembled into one binary file - PikeOS ROM image, which is then booted and run on the target system [25].

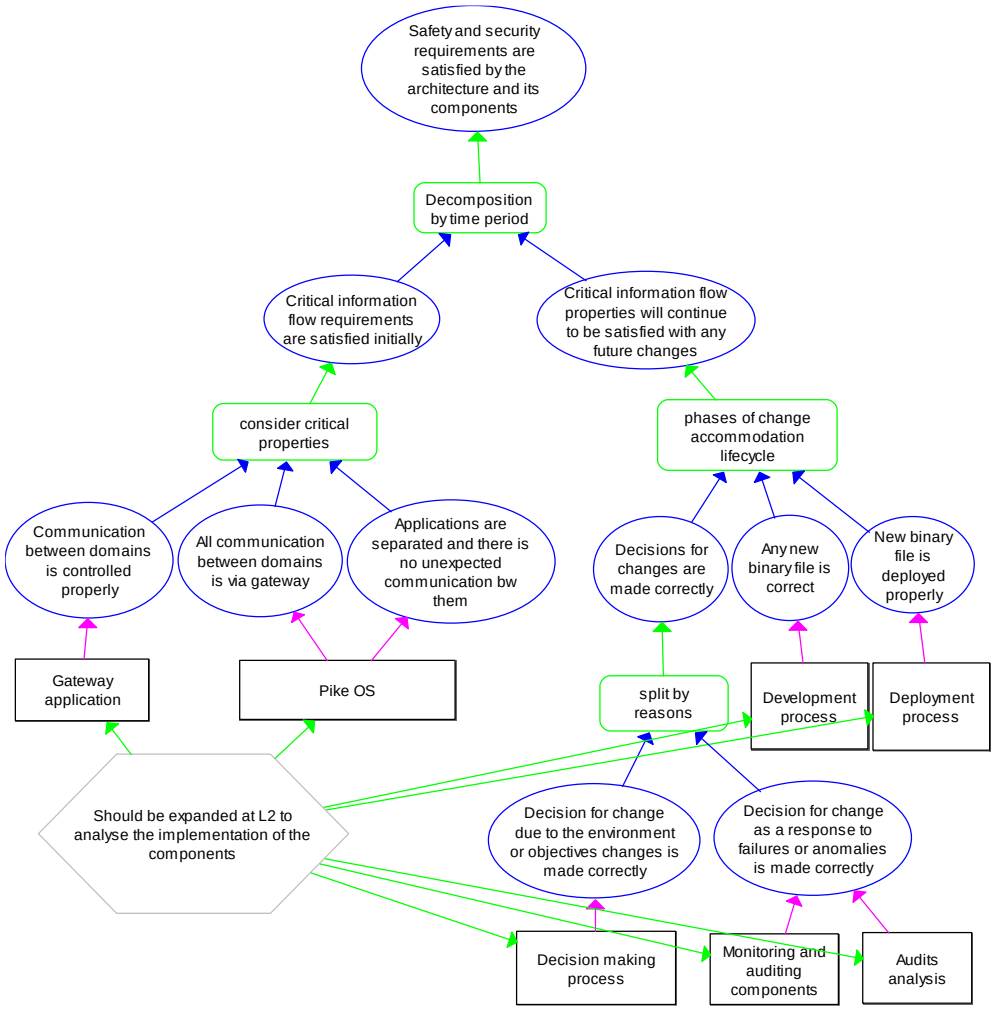

Figure 6. L1-level case fragment for the gateway

Therefore, the case created at the L2 level of abstraction includes (but is not limited to) the following claims:

1. Gateway is implemented and deployed as user applications within PikeOS partitions;
2. Inter-partition channels in the VMIT file are configured in a way that all partitions with different security settings can only communicate with the gateway partitions (inbound or outbound);
3. There are no errors in the configuration file allowing partitions with different se-curity settings to communicate bypassing the gateway partitions;
4. Partitions can only communicate by using the communication channels provided by PikeOS (e.g. via shared memory resources, network resource, etc.)
5. All PikeOS binary objects, configuration binary, gateway application binary and the resulting ROM binary image are generated properly without any malicious modifications or corruptions;
6. The binary objects are not modified or replaced after they have been generated.
7. All partitions are initialized, created and set up correctly before data passes the gateway;

8. Gateway application is loaded properly after the separation kernel has been established, and is available to use;

9. Gateway application can receive and send messages to the gateway partitions' ports, which are the end points for the communication channels configured in the VMIT file;

10. PikeOS security kernel correctly enforces any settings specified in the VMIT configuration file.

A fragment from the L2 case (related to #5 and 6 above) is provided in Figure 7.

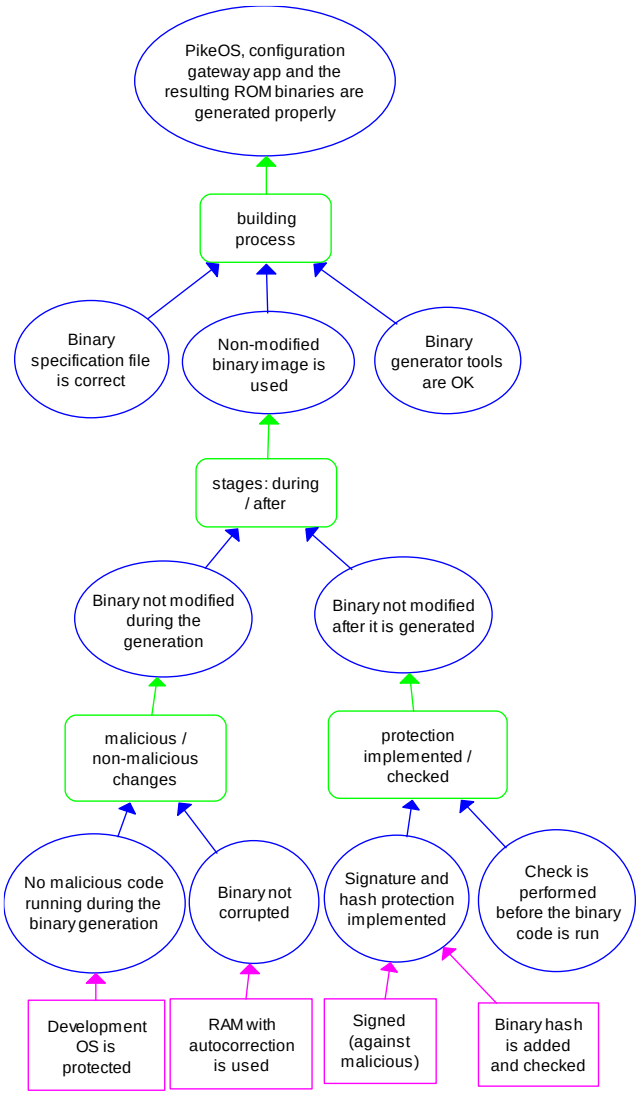

Figure 7. Sample fragment from the L2 case for the gateway

As discussed, various parts of the system need to be analysed within the L2 case. Other points (e.g. #4, 6, 7, 10) are related to the general PikeOS implementation and can be satisfied by showing that the PikeOS is implemented correctly to its specification providing all the required functionality. Some other claims are dependent on the correct implementation of the gateway application (#1, 9), or the experience of the system integrator (#2, 3), trusted development and deployment processes (#5, 6).

4 Conclusion

This paper presents a Layered Assurance approach to creating security-informed safety cases and its application to a MILS-based security gateway. We have developed CAE structures for each of the abstraction levels providing a concrete example of the approach that can be used to include security in a safety case as well as to consider how more general security issues can be addressed. There are two main roles of the approach: 1) it aids in the communication by providing a summary of the issues and their interrelationship, and 2) it indicates how we might reason that the lower properties combine to satisfy the top-level claim.

Overall, our review of the Layered CAE Assurance Case has broadened our view of how to combine safety and security. For example since tackling at architecture level is insufficient, we need to escalate to requirements using the abstraction layers and the explicit consideration of policy interactions within the L0 layer. In addition the consideration of lifecycle issues, particularly the adaptation and updating of the system is part of our assurance case approach.

The CAE Layered Approach provides a generic link between a number of key processes: the integrated risk analysis and the safety and security system development lifecycle, and further integration could be developed. We also found that the IMA architecture and PikeOS have intrinsic properties of separation and partitioning that are fundamental to enforcing the safety and security properties. While many of the assurance required from a security-informed safety perspective will exist in the IMA assurance, the emphasis on the credibility of the supply chain, the trust in tools, the response to

malicious events, maintenance and update policies will be different.

To show that the claims are a complete set and that the PikeOS and Security Gateway properties do in fact combine in this way will require us to provide a more formal semantics. One way to do this is to take a more explicit model based approach where the claim structure helps us identify the right level of abstraction and detail in the model. This is a topic for future research.

The next steps will be towards additional formalisation of the reasoning within the security-informed safety cases and the development of templates using the CAE Blocks [4] as well as exploring their linking them to formal models. Additionally, issues related to compositionality and traceability between layers would need to be addressed in more detail.

References

[1] ASCAD: Adelard Safety Case Development Manual. Adelard (2010)

[2] Toulmin, S.: The Uses of Argument. Cambridge University Press (1958)

[3] Kelly, T.: The goal structuring notation-a safety argument notation. In: Proc. DSN 2004, Workshop on Assurance Cases, 2004.

[4] Bloomfield, R. and Netkachova, K.: Building Blocks for Assurance Cases. 2nd International Workshop on Assurance Cases for Software-intensive Systems (ASSURE), International Symposium on Software Reliability Engineering, Naples, Italy (2014)

[5] ISO/IEC 15026-2 Systems and software engineering -- Systems and software assurance -- Part 2: Assurance case. ISO (2011)

[6] Delong, R.: Compositional Certification Lecture Notes. Real-Time Embedded Systems Forum, The Open Group (TOG) conference, Toronto, Canada (2009)

[7] Boettcher, C., Delong, R., Rushby, J., Sifre, W.: The MILS Component Integration Approach to Secure Information Sharing. 27th IEEE/AIAA Digital Avionics Systems Conference (DASC), St. Paul MN (2008)

[8] ISO 26262: Road Vehicles - Functional Safety. International Organization for Standardization.

[9] Railway applications - Communication, signalling and processing systems - Safety-related electronic systems for signaling. CSN EN 50129 (2003)

[10] Def Stan 00-55: Requirements for safety related software in defence equipment. Ministry of Defence

[11] Bloomfield, R., Netkachova, K., Stroud, R.: Security-Informed Safety: If it's not secure, it's not safe. 5th International Workshop on Software Engineering for Resilient Systems (SERENE), Kiev, Ukraine (2013)

[12] ED-135/ARP4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment. EUROCAE/RTCA (1996)

[13] ED-79A/ARP4754A: Guidelines for Development of Civil Aircraft and Systems. EUROCAE/RCTA (2010)

[14] ARINC-653: Avionics Application Software Standard Interface. Airlines Electronic Engineering Committee (2010)

[15] ED-202a/DO-326a: Airworthiness Security Process Specification. EUROCAE/RCTA (2014 and 2015)

[16] ED-203: Airworthiness Security Methods and Considerations. EUROCAE (2012)

[17] Boettcher C., R. Delong, J. Rushby, S. Wilmar: The MILS Component Integration Approach to Secure Information Sharing. 27th Digital Avionics Systems Conference (DASC), St. Paul, MN (2008)

[18] The Layered Assurance Workshop (LAW). ASCAS 2007-2013.

[19] DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations. RTCA (2005)

[20] ARINC 811: Commercial Aircraft Information Security Concepts of Operation and Process Framework. Airlines Electronic Engineering Committee (2005)

[21] D11.1 - Project Requirements: Classification, Cross-domain analysis and High-Level Architecture. EURO-MILS project (2014)

[22] PikeOS Manual - v3.4. SYSGO AG (2014)

[23] EURO-MILS: The EURO-MILS Project. 2012-2016. European Union's 7th Framework Programme. ICT-318353. www.euromils.eu

[24] Microsoft Corporation. 2002. The STRIDE Threat Model. Online available: https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx Accesses: 2015-08-13

[25] Installing and Using PikeOS. SYSGO AG (2014)

Acknowledgements

This work was supported by the Artemis JU SESAMO project (project number 295354), the European Union's 7th Framework Programme project EURO-MILS (ID: ICT-318353) and the UK EPSRC funded Communicating and Evaluating Cyber Risk and Dependencies (CEDRICS) project which is part of the UK Research Institute in Trustworthy Industrial Control Systems (RITICS).

Email Addresses

Kateryna.Netkachova.2@city.ac.uk

Kevin.Mueller@airbus.com

Michael.Paulitsch@thalesgroup.com

R.E.Bloomfield@city.ac.uk