HighIntegritySystems

WITTENSTEIN

Performing the Threat Analysis and Risk Assessment (TARA) Analysis Out Of Context

ISO 21434 And Software Components

Issue 1.0 - 18 April, 2024

This document was prepared by WITTENSTEIN high integrity systems ltd for the Embedded World Conference 2024.

Copyright date as document date.

WITTENSTEIN high integrity systems
Americas: +1 408 625 4712
ROTW: +44 1275 395 600

email: sales@highintegritysystems.com
web: www.highintegritysystems.com

WITTENSTEIN

CONTENTS

Contents 2
List Of Figures 4
List Of Notation 4

CHAPTER 1 Introduction 5

CHAPTER 2 The ISO 21434 Standard 6
2.1 Threat Analysis and Risk Assessment 6
2.1.1 Asset Identification 7
2.1.2 Threat Scenario Identification 7
2.1.3 Impact Rating 7
2.1.4 Attack Path Analysis 7
2.1.5 Attack Feasibility Rating 7
2.1.6 Risk Value Determination 7
2.1.7 Risk Treatment Decision 7

CHAPTER 3 Applying The ISO 21434 Standard to a Generic Software Component 8
3.1 Off the Shelf Component 8
3.2 Component Out of Context 8

CHAPTER 4 Performing the SAFE**RTOS**® Security Analysis 9
4.1 Document Assumptions of Use 9
4.1.1 Vulnerability during the Boot Process 9
4.1.2 Processor Architecture 9
4.1.3 Physical Deployment 9
4.2 Define the Context and Scope of the Component 10
4.3 Perform the TARA Out Of Context 10
4.1.1 Asset Identification 10
4.1.2 Threat Scenario Identification 10

4.1.3 Impact Rating....11
4.1.4 Attack Path Analysis....11
4.1.5 AttackFeasibilityRating....11
4.1.6 RiskValueDetermination....12
4.1.7 Risk Treatment Decision....12

CHAPTER 5 Lessons Learned While Performing the SAFE**RTOS**® Security Analysis....13

CHAPTER 6 The SAFE**RTOS**® Enhanced Security Module....14
6.1 Obfuscated Handles....14
6.2 Access Control Policy (ACP)....15
6.3 Object Access Control Policy (OCAP)....15
6.4 Penetration Detection Monitor....15
6.5 Task Context Data Isolation....15
6.6 SAFE**RTOS**® to Task Data Isolation....15

CHAPTER 7 Conclusion....16

References....17

Contact Information....18

List of Figures

Figure 1 ISO 21434 Scope....6
Figure 2 Using ISO 21434 With Third Party....8
Figure 3 Performing the SAFE**RTOS**® Security Analysis....9
Figure 4 Defining the threat boundary in SAFE**RTOS**®....10
Figure 5 Summary of the TARA Process....11
Figure 6 Integration of Security, Safety and Functional Requirements....12
Figure 7 SAFE**RTOS**® and the ESM....14

List of Notation

ACP	Access Control Policy
API	Application Programming Interface
ESM	Enhanced Security Module
MMU	Memory Management Unit
MPU	Memory Protection Unit
OACP	Object Access Control Policy
OCB's	Object Control Blocks
TARA	Threat Assessment and Risk Assessment

CHAPTER 1 Introduction

This paper explores the application of ISO 21434 standards to enhance cybersecurity in embedded systems, focusing on the automotive industry. It discusses the integration of cybersecurity practices into the product life cycle, including conducting SAFE**RTOS**® security analysis and incorporating Threat Assessment and Risk Assessment (TARA) results. Insights and lessons learned from TARA implementation are presented, alongside the practical outcome: the SAFE**RTOS**® Enhanced Security Module. This module exemplifies the tangible benefits of aligning cybersecurity practices with industry standards, offering a structured approach to mitigate risks in interconnected vehicle components.

Cybersecurity is increasingly important in all aspects of engineering. The aspects of this which relate to IT infrastructure and social media are well known, however, this subject is rapidly gaining significance in embedded systems which were once considered suitably isolated from threats from the outside world. The expanding interconnectedness of vehicle components both internally and externally contribute to provide a complex operational environment and an increasing variety of attack vectors.

Within the automotive industry ISO 21434 [1] was published in 2021 to provide a framework and common language for communicating and managing cybersecurity risks. “Framework” and “Common Language” are very important terms, a vehicle is not a single system. It is made of many components produced by various suppliers. The major components themselves are made of subcomponents, again produced by a variety of suppliers. This model reflects down the supply chain, therefore a common language to quantify, mitigate and share risks is necessary.

In this paper, we will talk about:

- The ISO 21434 standard;
- Applying ISO 21434 to a generic software component;
- Performing the SAFE**RTOS**® security analysis;
- Feeding the results of the Threat Assessment and Risk Assessment (TARA) back into a Functional Safety product life cycle;
- Lessons learned while performing the TARA;
- The practical outcome of the process, the SAFE**RTOS**® Enhanced Security Module.

CHAPTER 2 The ISO 21434 Standard

ISO 21434 is concerned with analysing cyber security threats within road vehicles. All organisations involved in the supply chain must have adequate cybersecurity management. This applies to the organisation itself, as well as the product and is applicable throughout the product life; from concept to ongoing support operations as seen in Fig 1.

The standard addresses:

- Organisational cybersecurity management, how the company manages itself in relation to cybersecurity;
- Project dependent cybersecurity management, how the project manages cybersecurity;
- Distributed cybersecurity activities, management of suppliers etc;
- Continual cybersecurity activities, ongoing monitoring and incident management;
- Product Development phases, including concept, development and post development phases;
- Threat Analysis and Risk Assessment methods.

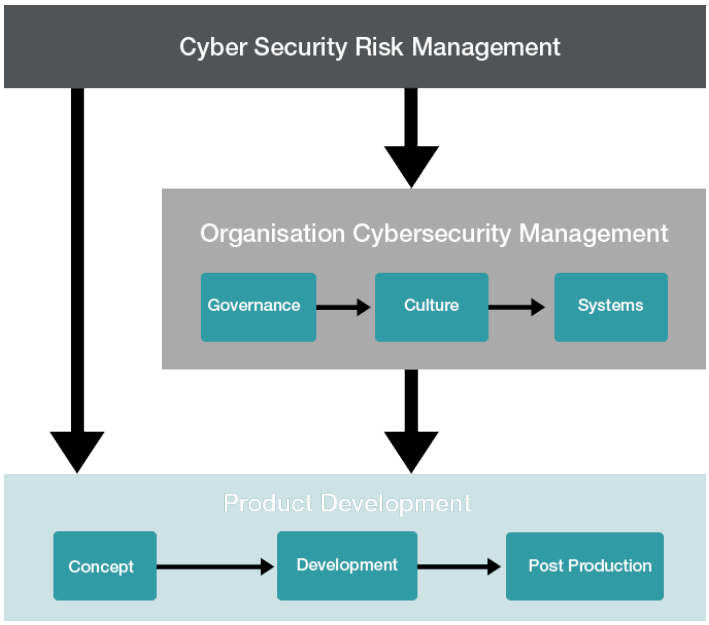

Figure 1 *ISO 21434 Scope*

Threat Analysis and Risk Assessment is more generally referred to as TARA and is referenced in many places throughout the standard as the output or identification of risks. The TARA is a continual activity and may require mitigations at organisational level, or project management level as well as directly influencing the product development.

2.1 Threat Analysis and Risk Assessment

The TARA process consists of a number of activities, the standard expresses them in general terms and does not require the use of any particular tools or methods.

2.1.1 Asset Identification
This is concerned with assets with cybersecurity properties which can lead to a damage scenario if they are compromised.

2.1.2 Threat Scenario Identification Asset Identification
Assets can have multiple cybersecurity properties; this activity is concerned with determining how a cybersecurity asset can be compromised and determining use cases.

2.1.3 Impact Rating
The effect of the identified threats shall be assessed against the categories of safety, financial, operational and privacy.

2.1.4 Attack Path Analysis
This activity is concerned with determining how the identified threats could be manifested.

2.1.5 Attack Feasibility Rating
Each attack path should be analyzed to determine how much effort is required to realize the attack and should consider access to specialized knowledge or equipment and access to the asset.

2.1.6 Risk Value Determination
Based on preceding steps, each risk is given a numerical rating.

2.1.7 Risk Treatment Decision
Each risk shall be evaluated for treatment, this includes:

- Avoiding the risk. Action is taken to eliminate the risk by removing the risk sources;
- Reducing the risk. Action is taken to reduce the risk likelihood or impact;
- Sharing the risk. Some risks can only be mitigated elsewhere in the supply chain;
- Retaining the risk. Sometimes a risk must be accepted.

CHAPTER 3 Applying the ISO 21434 Standard to a Generic Software Component

The good news is that ISO 21434 does recognise the existence of the supply chain and does attempt to address the applicability of the standard to off the shelf components.

Essentially, there are two paths to managing the use of a third-party component, it can be treated as a component out of context or directly as an off the shelf component. These are not mutually exclusive terms and as we will see, it is more to do with the way in which the integration is approached, as seen in Fig. 2.

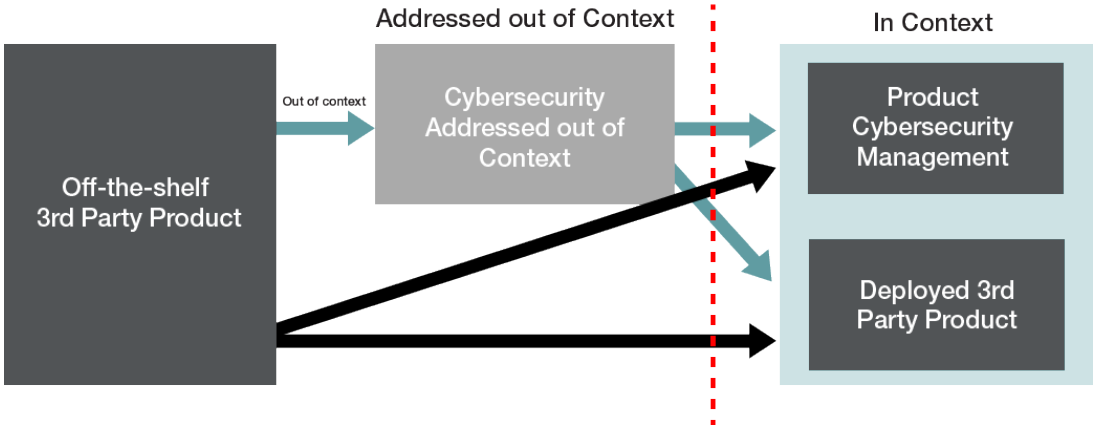

Figure 2 *Using ISO 21434 With Third Party*

3.1 Off the Shelf Component

When an 'off the shelf' is selected for use, the organization intending to use the component is responsible for determining whether the component is suitable for the intended use. This is a normal engineering use case where the user will determine whether the tool or component can "do the job".

When viewed from a cybersecurity perspective this involves reviewing documentation to determine whether allocated cybersecurity requirements can be fulfilled, determining whether the component is suitable for use in its intended context and whether the existing documentation is sufficient to support ongoing cybersecurity perspectives.

If the existing documentation is insufficient to achieve the required cybersecurity goals, the activities needed to conform with ISO 21434 need to be identified and performed.

3.2 Component Out of Context

Where a component is being developed without a specific application in mind, it is viewed as being developed 'out of context'. Again, this is a normal engineering use case where a supplier will produce a piece of equipment with the intention that is can be deployed in many applications.

Where this differs from the 'off the shelf' model is that the supplier is aware of the cybersecurity responsibilities and has documented the assumptions of the intended use, its intended deployment context and any external interfaces.

In this case, the supplier shall generate and implement cybersecurity requirements based on the documented assumptions. When deployed in context the cybersecurity claims are validated.

CHAPTER 4 Performing The SAFERTOS® Security Analysis

For a component like SAFE**RTOS**®, there are problems due to the lack of a tangible external interface in the context of an end-user application. An RTOS provides a set of services to an application which are controlled by the API. If the component supplier does not know the application's purpose, how it will be deployed or the consequences if the component is compromised?

A security analysis of the SAFE**RTOS**® API had been performed in the past and some weaknesses had been identified, these results were recorded and formally classified as part of the TARA process, portrayed in Fig. 3.

Figure 3 *Performing the* SAFE**RTOS**® *Security Analysis*

4.1 Document Assumptions of Use

With an out of context product, some assumptions are necessary otherwise we cannot determine the threat boundary let alone produce threats or solutions. For SAFE**RTOS**®, the assumptions related to the capabilities and configuration of the underlying processor and the place that SAFE**RTOS**® fills within the overall system.

4.1.1 Vulnerability during the Boot Process

The scheduler is initialized and started by the host application once it starts running. The deployment model is that SAFE**RTOS**® is linked directly to the host application. This means that SAFE**RTOS**® cannot mitigate any threats related to the boot process and therefore it was assumed that a suitable cryptographic boot process is being employed. If the firmware is compromised at source, then a software component cannot mitigate the threat.

4.1.2 Processor Architecture

For an RTOS to provide any partitioning or protection in a security environment, a Memory Protection Unit (MPU) or Memory Management Unit (MMU) is required. It was assumed that this was present, supported by SAFE**RTOS**® and was configured according to the SAFE**RTOS**® safety manual.

In a related manner, we assumed that the host architecture supports at least two privilege levels: privileged and unprivileged. Privileged allows access to all data regions, not just the ones associated with a task.

4.1.3 Physical Deployment

Finally, as SAFE**RTOS**® is deployed as part of an unknown application in an unknown hardware environment, it was assumed that all threats relating to direct physical access are outside the scope of our component.

Interestingly, following completion of the initial TARA process it was felt that these assumptions should be documented in the TARA and classified as 'Shared' requirements, as it is necessary for an integrator to mitigate them.

4.2 Define the Context and Scope of the Component

The key to this part of the process is determining the threat boundary. As we have a generic software component, it's not appropriate to phrase this in terms of communication busses or external interfaces even though these form the likely attack vectors. Instead, it was determined that the threat boundaries were the interfaces between the SAFE**RTOS**® kernel and the host application, Fig.4. In some cases, this related to application code interfacing with the RTOS via hook functions or the API. Whereas, in other cases, the interface is formed when accessing data objects that both the kernel and application must access.

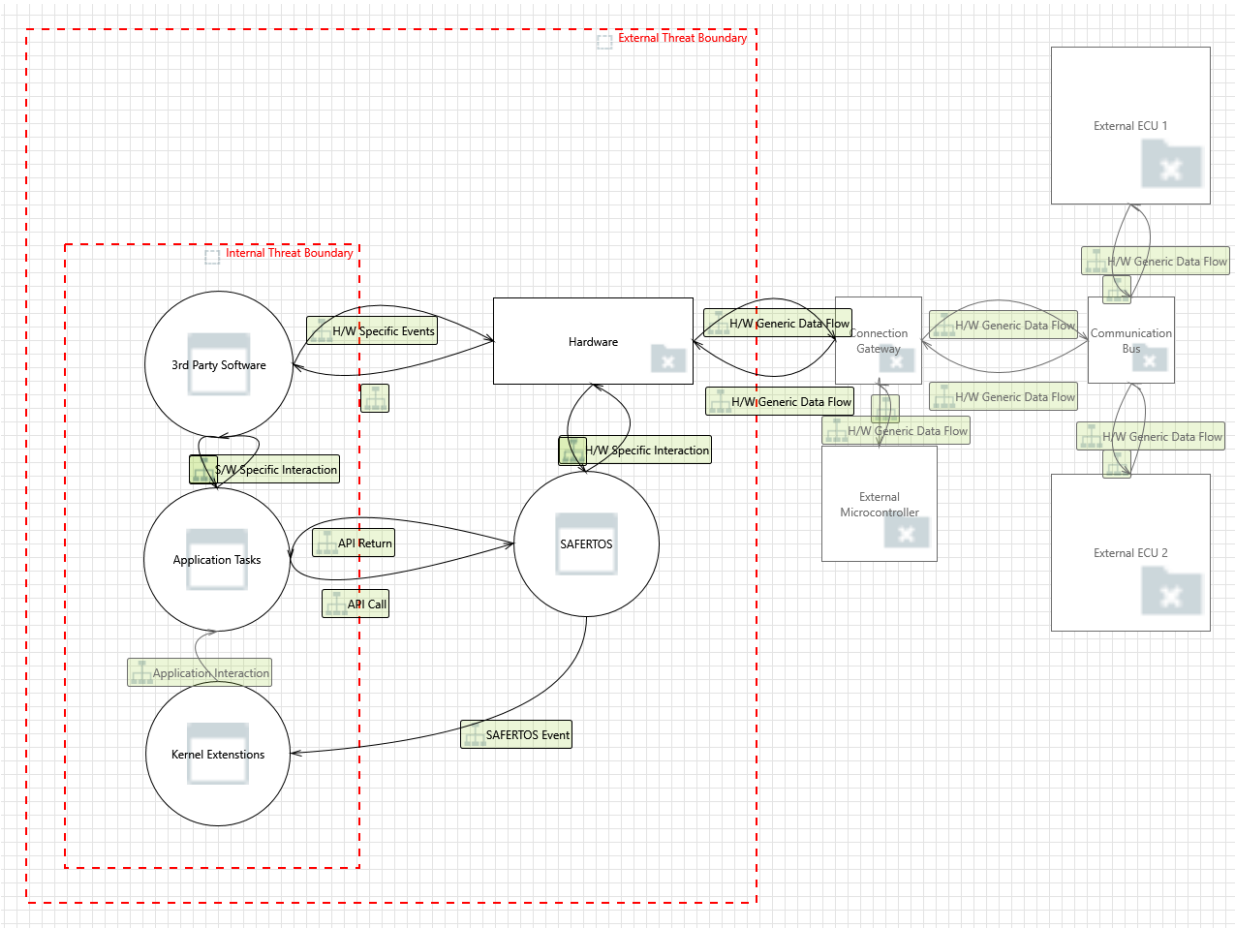

Figure 4 *Defining the threat boundary in* SAFE**RTOS**®

4.3 Perform the TARA Out Of Context

4.3.1 Asset Identification

For us the interfaces across the threat boundary were deemed to be the assets. In the security context diagram, Fig.5, each interface across the threat boundary was considered as part of the TARA exercise.

4.3.2 Threat Scenario Identification

Assets can have multiple cybersecurity properties, several frameworks exist. We selected the STRIDE model (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege). Each asset was considered for each of the threat scenarios.

4.3.3 Impact Rating

When considering a general purpose element such as SAFE**RTOS**® out of context, it is not possible to make judgements regarding the damage scenario – the 'Safety', 'Financial', 'Operational' or 'Privacy' consequences of an attack can only be properly assessed with knowledge of the application. Once a bad actor has access, nearly any attack path that allows a rogue task to gain privilege, will allow major degradation to the system.

As the category of damage cannot be determined out of context, all risks are categorised as affecting all categories.

4.3.4 Attack Path Analysis

Each of the threat scenarios were considered to determine if the attack was feasible. This is a key input to determining the actual feasibility of an attack succeeding.

4.3.5 Attack Feasibility Rating

Five attributes of possible attack paths (specialist knowledge, window of opportunity, equipment/effort, elapsed time and knowledge of the item) are ascribed numerical values and these are combined in order to determine a banded rating for attack feasibility.

4.3.5.1 'Specialist Expertise'

It is assumed that there are no firewalls or secure communications etc. Consequently, it would require "Layman" level knowledge of the full system to navigate whatever comms interface was used to implant a rogue task or tamper with the host memory.

4.3.5.2 'Window of Opportunity'

The extent of the connection to the external environment (ranging from wired CANbus to wireless and remote) is unknown. Therefore, the Window of Opportunity must be considered as "Unlimited" i.e. with high availability to the component with unlimited physical or logical access.

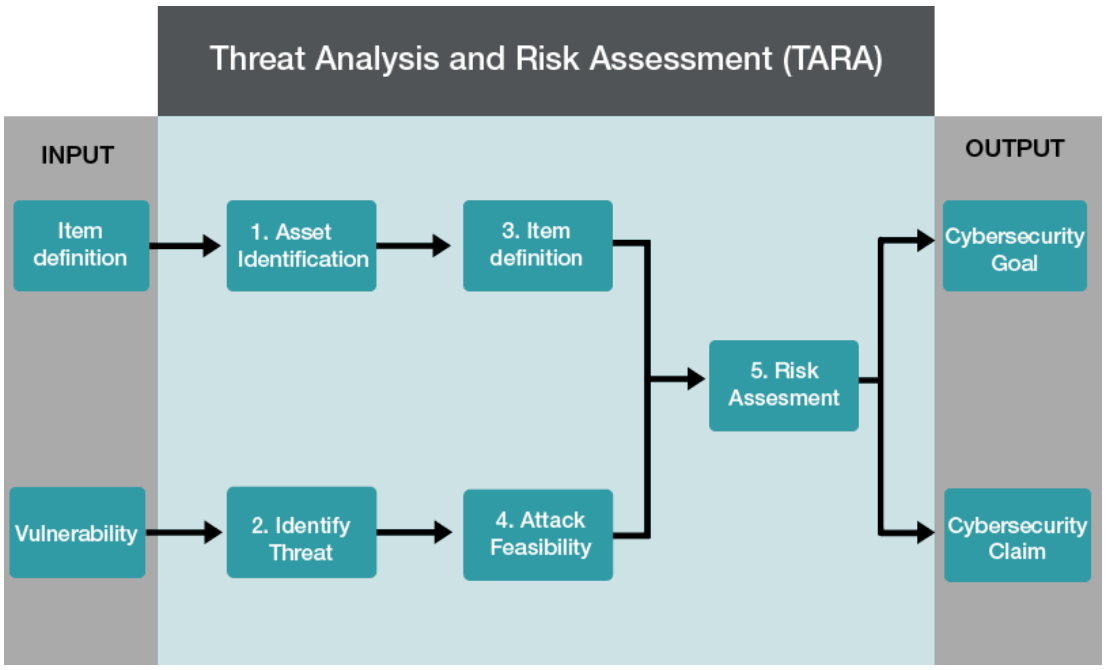

Figure 5 *Summary of the TARA Process*

4.3.5.3 'Equipment / Effort'

The unspecified nature of the system being scrutinised means the equipment to access the interfaces will be assumed to be "Standard" – a cell phone or laptop, but running specialised software developed for the purpose of hacking (exhaustive searches, dictionary attacks).

4.3.5.4 'Elapsed Time'

The time to develop a method cannot be assumed to take any reasonable effort, therefore the Elapsed Time is to be "≤1 day".

4.3.5.5 'Knowledge of the Item'

Whilst documents produced by WHIS are always restricted, it cannot be assumed that the integrating party has any document control measures. Therefore, the knowledge of the item must be considered to be freely available to anyone (e.g. found online, or inherent in the system) i.e. "Public".

Given the reported frequency of successful cybersecurity attacks on vehicles, and the ease with which the steps in these attacks can be replicated once discovered, it is agreed to be best practice to set all criteria to their worst case, arriving at a more realistic forecast of susceptibility.

4.3.6 'Risk Value Determination'

The impact of a potential risk occurring is infinite, as without knowing the scope and context of the system, any and all damage is possible. All impact due to the threats considered under all categories are deemed "Severe" - the maximum possible value - on this basis.

4.3.7 'Risk Treatment Decision'

Based upon each of the identified risks, a treatment decision was made:

Many of the risks were identified as 'Shared', since the risk could only be mitigated by the way in which SAFE**RTOS**® was deployed and configured.

Many of the risks were categorised as 'Reduce', as we felt that the risk could be mitigated by providing extra safeguards in the product.

Some of the risks had to be accepted as the ability for the host application to introduce privileged code into the system could not be entirely mitigated.

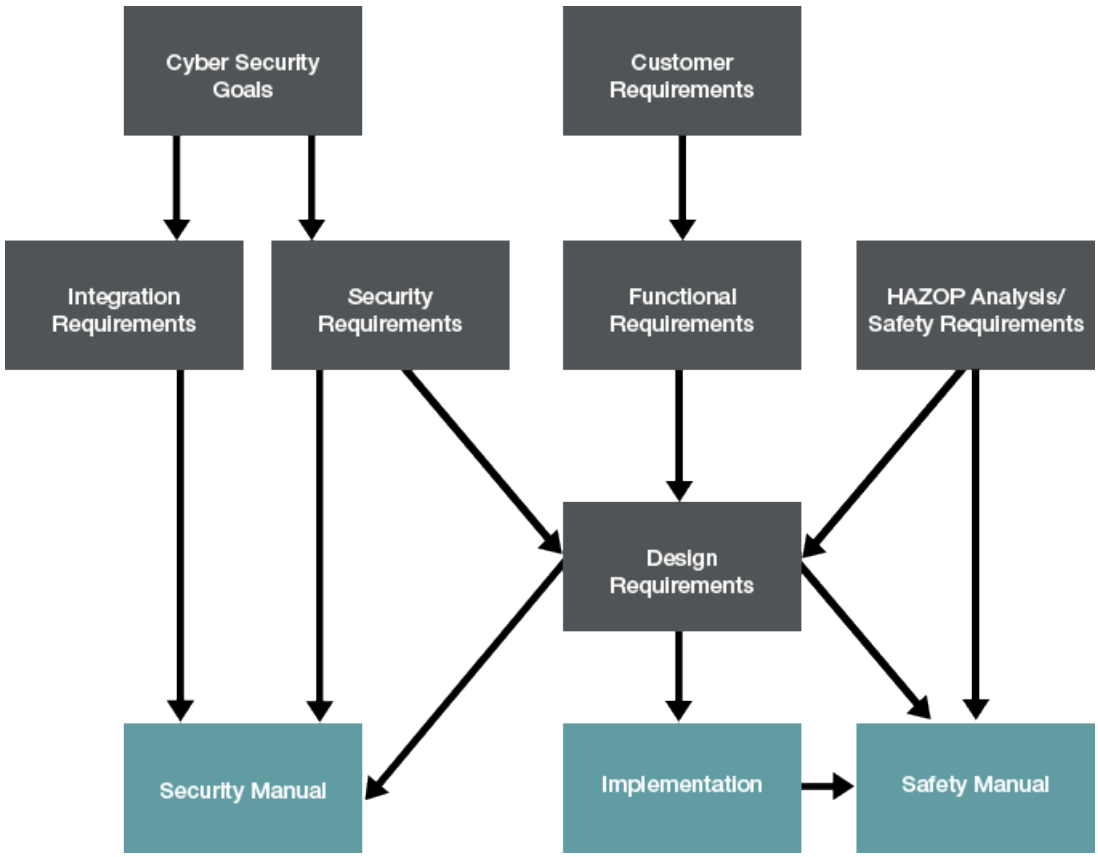

Figure 6 *Integration of Security, Safety and Functional Requirements*

CHAPTER 5 Lessons Learned While Performing The SAFE**RTOS**® Security Analysis

The hardest parts of this process were those that came at the beginning:

- Attempting the TARA process without an agreed definition of the context was impossible. If developing out-of-context, a 'typical' automotive use case should be agreed upon to allow reasonable identification of assets and possible threats. The context presented in this paper was a result of several false starts, but once it was in place the assets and threats could be identified;

- Asset definition can be difficult, too high a level does not allow for identification of all threats. Too much focus on the physical assets can also be misleading when considering an element out of context;

- With an out of context element, determination of 'Attack Feasibility', 'Impact Rating' and 'Risk Value' is fraught. We made the decision to assume the worst, but this also has issues as it can lead you to determine that an unrealistic amount of penetration and fuzz testing is required;

- For an 'Out of Context' element, we can only mitigate to a certain extent. There will be 'Shared' threats that can only be mitigated by the host application taking action or using and configuring the product in an approved manner. Our objective is to make this as clear as possible.

CHAPTER 6 The SAFERTOS® Enhanced Security Module

The large part of the outcome of our TARA process was the SAFE**RTOS**® Enhanced Security Module (ESM). Even when memory protection schemes are in force using an MMU or MPU, the nature of the API is such that a corrupted Task can easily compromise the entire system. To this end, SAFE**RTOS**® is now available with an Enhanced Security Module to give the application developer more tools to restrict the behaviour of individual tasks, Fig.7.

The SAFE**RTOS**® ESM relies on hardware support via the MPU or MMU to enforce the allocation of data and code to specific regions. It therefore permits Task to Task and Task to kernel spatial separation. The SAFE**RTOS**® ESM consists of a high-level API wrapper module which enforces Object Handle Obfuscation, an API Access Control Policy, and Object Access Control Policies to the SAFE**RTOS**® API. It also requires a security aware portable layer to ensure security of Task context data and to provide the tightest API boundary between Task and kernel execution. These elements work together to provide a robust and secure real-time operating system.

The ESM alone will not make an embedded application secure. It is intended to be used alongside other security protection mechanisms such as secure booting (root of trust), security keys and encryption of communication channels, as well as physical security techniques driven by hardware design or physical usage limitations.

6.1 Obfuscated Handles

In SAFE**RTOS**® all RTOS features such as Tasks, Queues, Timers etc. have an associated structure that provides SAFE**RTOS**® with the information required to manage the instance of the feature correctly. These are typically referred to as Object Control Blocks (OCB's) and the host application interfaces with these objects via an Object Handle. In standard SAFE**RTOS**® implementations, the Object Handle is usually just a 'C pointer' to the OCB. When using the ESM, Object Handles differ in that they do not contain a direct reference to the OCB for the created object. Instead, they contain an obfuscated number which has no direct link to the object in question.

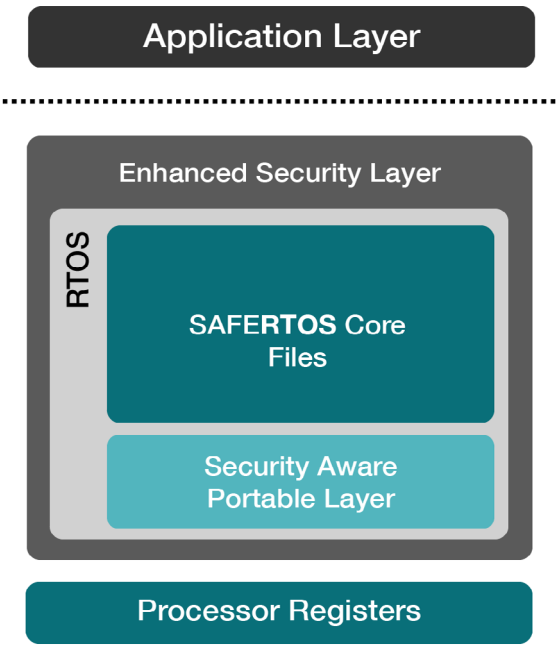

Figure 7 SAFE**RTOS**® *and the ESM*

Internally, the ESM maintains a lookup table which allows fast and deterministic conversion of the Object Handle to find the referenced OCB whenever an API function is called. The time complexity of the lookup operation is always constant, regardless of the number of Handles in use.

The total number of Handles required by the application must be determined and defined as part of the system configuration. Handles of objects that are deleted can be re-used for newly created objects once the scheduler has started.

6.2 Access Control Policy (ACP)

The Access Control Policy (ACP) allows each Task to be configured with a set of permissions to allow access to a specific subset of the SAFE**RTOS**® API functions. Once configured, the Task will only be able to successfully access those functions that have been specified: attempting to access other functions will result in an error code being returned and no other actions will be performed.

When Tasks are created prior to the SAFE**RTOS**® scheduler being started, they are granted user specified default permissions. The default permissions can then be overridden using xTaskConfigureACP() prior to the scheduler starting.

This allows the user to have complete control of API access configuration while applying strict usage restrictions when the system is running.

6.3 Object Access Control Policy (OACP)

The Object Access Control Policy (OACP) allows a Task to be configured so that it has access to a limited set of other SAFE**RTOS**® Objects (i.e. Timers, Queues, Mutexes, Semaphores, Event Groups, Event Polls and other Tasks). Once configured, the Task will only be able to successfully access those Objects that have been specified using their Handle: attempting to use any other Object's Handle will result in an error being returned and no other actions will be performed.

6.4 Penetration Detection Monitor

The ESM provides an option to implement a centralised error hook function which allows an application to intercept all result codes returned from all SAFE**RTOS**® API functions. The intended use for this is to allow an application to detect whether a Task or Tasks are generating unexpected error results when calling the SAFE**RTOS**® API functions, which may indicate either a system failure or a hacking attempt (e.g. a 'fuzzing' attack).

6.5 Task Context Data Isolation

When using the ESM, the actual Task context data is stored in the TCB (Task Control Block) rather than the Task stack. The enhanced memory map requirements force the TCB's to be located in memory that is not accessible to user Tasks. This protects the inactive Tasks from exploitation and data leakage due to access by a different Task.

6.6 SAFE**RTOS**® to Task Data Isolation

All data transfer between SAFE**RTOS**® and user memory spaces is checked for validity before the access is performed. All control transfer between SAFE**RTOS**® and user space (API calls) is performed as a clean uninterruptable operation.

For more information, please refer the ESM Datasheet [Reference 3] and the RTOS Security: SAFE**RTOS**® and Its Enhanced Security Module Whitepaper [4].

CHAPTER 5 Conclusion

This paper has examined the steps taken to implement an extension to SAFE**RTOS**® to enhance capabilities when deployed in a cybersecurity sensitive context. It has considered the relevant standards, the application of the standard to a particular product that needs to be deployable as a general purpose. Components, in devices, that themselves are only part of the vehicle supply chain. It has considered the issues relating to incorporation of requirements that need to be mitigated outside the product itself, and the creation of the product enhancement itself.

For more information, please refer to www.highintegritysystems.com.

References

[1] ISO 21434:2021, Road Vehicles - Cybersecurity Engineering.

[2] ISO 26262-6:2018, Road vehicles - Functional safety Part 6: Product development at the software level.

[3] WITTENSTEIN high integrity systems, “Enhanced Security Module Datasheet”.
https://www.highintegritysystems.com/down-loads/manuals-datasheets/safertos-datasheet-downloads/

[4] WITTENSTEIN high integrity systems, “RTOS Security: SAFE**RTOS**® and Its Enhanced Security Module Whitepaper”.
https://www.highintegritysystems.com/downloads/white_papers/ESM_Security_whitepaper.pdf

WITTENSTEIN

CONTACT INFORMATION

User feedback is essential to the continued maintenance and development of SAFE**RTOS**®. Please provide all software and documentation comments and suggestions to the most convenient contact point listed below.

Contact WITTENSTEIN high integrity systems

Address: WITTENSTEIN high integrity systems
Brown's Court, Long Ashton Business Park
Yanley Lane, Long Ashton
Bristol, BS41 9LB
England

Phone: +44 (0)1275 395 600
Email: support@HighIntegritySystems.com

Website www.HighIntegritySystems.com

All Trademarks acknowledged.

This document was prepared by WITTENSTEIN high integrity systems ltd for the Embedded World Conference 2024.

Copyright date as document date.

Copyright 2024 WITTENSTEIN high integrity systems Ltd. V1.3. Correct at time of issue.