



TRACE

inTegration & haRmonizAtion
of logistiCs opERations

D4.5 Security scheme and Visual interfaces (A)

Horizon Innovation Actions | Project No. 101104278

Call HORIZON-CL5-2022-D6-02



Co-funded by
the European Union

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor CINEA can be held responsible for them

Dissemination level	Public (PU)
Type of deliverable	R – Document, report
Work package	WP4 – Infrastructure & Ecosystem
Status - version, date	Final – 1.1, 27/11/2024
Deliverable leader	CSEM
Contractual date of delivery	30/11/2024
Actual date of delivery	30/11/2024

List of authors

Author Name	Organization
Damian Vizár	CSEM
Lucas David Meier	CSEM
Alessio Masola	UNIMORE
Paolo Burgio	UNIMORE
Andronikidis Georgios	SID
Niotis Georgios	SID
Spyridon Thermos	CDW
Anargyros Chatzitofis	CDW
Nikolaos Zioulis	CDW
Nikolaos Tymplalexis	UNIS
Peggy Papadopoulou	UNIS
Tejas Bhagat	BC5

Paola Lorenzoni	ISIG
-----------------	------

Version History

Version	Date	Author	Description of changes
1.0	07.11.2024	D. Vizár	Review-ready (previous versioning continuous in TRACE SharePoint).
1.1	26.11.2024	D. Vizár	Camera-ready (internal reviewers' comments integrated).

Peer Review

	Reviewer Name	Organization	Date
	Ioanna Mesogiti	OTE (ex. COSMOTE)	
	Theofilos Triommatis	TUC	

Quality Manager Review

	Reviewer Name	Organization	Date

Legal Disclaimer

The information in this document is provided “as is”, and no guarantee or warranty is given that it is fit for any specific purpose. The TRACE project Consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

Executive Summary

This deliverable aims at addressing two primary areas within the project's scope: security and privacy measures (Task 4.6), as well as visual interface design (Task 4.7). Security requirements are central to the report, as they significantly influence the architecture and design of the platform's visual interfaces. This report focuses on the specification and preliminary implementation aspects, highlighting the efforts to meet these standards in a phased approach across two releases (scheduled for M18).

In the first part of the report, a threat analysis identifies relevant threats and potential vulnerabilities for TRACE. The methodology derived from the NIST SP 800 30 consists in a system characterization, threat event identification and vulnerability mapping. The system characterization includes an inventory of assets, as well as a connection-flow diagram, which models boundaries, component interactions and possible entry points. The threat identification strategy is to focus on high-level threats that impart a clear intuition on the impact. The vulnerability mapping then links these threats to low-level technical issues that might enable an attack. The threat analysis output is a list of security controls, selected to mitigate the identified vulnerabilities. The threat analysis and the list of controls can be used as an input for a risk assessment by the future adopters of TRACE, as well as a guide for implementation of extensions. A subset of these controls is selected as necessary even for the demonstrators, to start a discussion towards ensuring a sufficient level of protection for the latter. Finally, security and privacy technologies being developed as contributions in Work package 4, which correspond to a subset of the identified controls, are described in detail, their security aspects characterized, and integration considerations discussed. These are secure boot and firmware update for vehicle-to-vehicle communication module, vehicle-to-vehicle communication security, secure data collection with Bike Connection Box, blockchain SSID module, intrusion Detection Module and Events Management, and end-to-end encryption of data in transit and at rest.

The second part of the deliverable is dedicated to the Virtual Cockpit, an advanced visual interface for unmanned vehicles. The Virtual Cockpit supports high-resolution, real-time video streaming, localization, user friendly interface and a number of emergency controls. The data streams and security features underlying the Virtual Cockpit are discussed in detail.

Table of Contents

Executive Summary.....	4
Table of Contents	5
Table of figures.....	8
Table of tables.....	9
Definitions, Acronyms and Abbreviations.....	10
1 Introduction	12
1.1 Scope of deliverable	13
1.2 Relation with other work packages/deliverables	14
1.3 Intended audience.....	14
1.4 Deliverable structure	15
2 TRACE Security and Privacy Framework and Implementation Aspects	16
2.1 TRACE Threat analysis.....	16
2.1.1 System characterization	16
2.1.2 Threat Sources.....	26
2.1.3 Threat events.....	27
2.1.4 Vulnerabilities.....	35
2.2 Security controls for TRACE architecture.....	44
2.2.1 Controls compulsory for demonstrators	54
3 Security controls developed in TRACE	55
3.1 Secure boot and firmware update for V2V communication module	55
3.1.1 Specification	56
3.1.2 Security considerations	57
3.1.3 Integration considerations	58
3.2 V2V communication security.....	59

3.2.1	Specification	60
3.2.2	Security considerations	62
3.2.3	Integration considerations	63
3.3	Secure data collection with Bike Connection Box.....	63
3.3.1	Specification	64
3.3.2	Security considerations	64
3.3.3	Integration considerations	64
3.4	Blockchain SSID module	65
3.4.1	Specification	65
3.4.2	Security considerations	66
3.4.3	Integration considerations	68
3.5	Intrusion Detection Module and Events Management	69
3.5.1	Specification	70
3.5.2	Security considerations	74
3.5.3	Defeating Identified Attacks	74
3.5.4	Effective Control	75
3.5.5	Integration considerations	75
3.5.6	Integration into TRACE Architecture	76
3.6	End-to-end encryption of data in transit and at rest.....	78
3.6.1	Specifications.....	79
3.6.2	Security Considerations.....	83
3.6.3	Integration Considerations	84
4	Virtual Cockpit.....	87
4.1	Description of the component.....	88
4.1.1	Key Features	88

4.2	Virtual Cockpit Data Streams.....	90
4.3	Security & Privacy Options	91
5	Conclusion.....	92
6	References.....	93

Table of figures

Figure 12-2 TRACE reference architecture..... 18

Figure 32-4 Interpretation of the symbols used in connection flow diagram..... 20

Figure 52-6 TRACE architecture 21

Figure 72-8 TRACE connection flow diagram for threat modelling..... 22

Figure 93-10 Secure boot and FUOTA from V2V communication module. The component will be developed in two iterations. In the first iteration (green), an automatic FUOTA for vehicles with direct Internet connection (e.g., 5G) is implemented. In the second iteration, the support for an automatic propagation of the update to vehicles that may not have Internet connection (e.g., for a more cost-effective fleet with platooning) is added (blue). 56

Figure 113-12 Ad-hoc group key distribution for V2V communication in TRACE. Here an example with threshold $T=3$, with V2V modules identified with IDs “1”, “3” and “7”. 61

Figure 133-14 Securing communication in a group of V2V modules after key distribution in Figure 113-12. 62

Figure 4-1: Overview of the Virtual Cockpit Module in TRACE platform 88

Table of tables

Table 1 TRACE assets and their security objectives	23
Table 2 Threat/Attack sources for TRACE	26
Table 3 TRACE threat events	28
Table 4 TRACE vulnerabilities	36
Table 5 TRACE security controls	44

Definitions, Acronyms and Abbreviations

Abbreviation	Definition
3PLs	Third-Party Logistics Providers
4PLs	Fourth-Party Logistics Providers
AES	Advanced Encryption Standard
API	Application Programming Interface
APT	Advanced Persistent Threat
CI/CD	Continuous Integration/Continuous Deployment
CINEA	European Climate, Infrastructure and Environment Executive Agency
CPU	Central Processing Unit (CPU)
CSRF	Cross-Site Request Forgery
DEK	Data Encryption Key
DLT	Distributed Ledger Technology
DoS	Denial of Service
FUOTA	Firmware Update Over The Air
GDPR	General Data Protection Regulation
GPS	Global Positioning System
HSM	Hardware Security Module
IAM	Identity and Access Management
KEK	Key Encryption Key
KES	Key Encryption Service
MCU	Microcontroller Unit
RBAC	Role-Based Access Control
SHA	Secure Hash Algorithm
SPI	Serial Peripheral Interface
SSID	Service Set Identifier
TLS	Transport Layer Security
UART	Universal Asynchronous Receiver-Transmitter
UX	User experience
V2I	Vehicle-to-Infrastructure

Abbreviation	Definition
V2V	Vehicle-to-Vehicle
VR	Virtual Reality
w.r.t.	with respect to
WAF	Web Application Firewall
XSS	Cross-Site Scripting
UX	User Experience

1 Introduction

This deliverable presents the security, privacy, and advanced visual interfaces activities undertaken within the project, which aims to develop a robust platform for the collection, processing, and sharing of data in the logistics and delivery sector.

In the context of **cyber security**, the focus of this report is twofold: firstly, it provides a comprehensive threat analysis and defines the necessary security controls; secondly, it delves into the detailed specification and discussion of several concrete security technologies implemented by project partners. The logistics and delivery sector is characterised by a complex, multi-stakeholder environment where data sensitivity and the risk of data spills to competitors pose significant concerns. The TRACE platform must thus address these challenges by ensuring robust security measures that protect against unauthorised access and data breaches. On the other hand, the development on advanced, (semi-)autonomous vehicles requires robust protection of on-vehicle features, vehicle to vehicle communication and monitoring of vehicle security status.

The advanced security technologies developed in TRACE are:

- **Secure Boot and Firmware Update for Vehicle-to-Vehicle Communication:** This technology ensures that only authenticated firmware is executed on constrained embedded platforms, enhancing the security of vehicle-to-vehicle communications.
- **Secure Communication and Ad-Hoc Group Key Establishment:** Utilizing threshold cryptography, this approach secures vehicle-to-vehicle communications by establishing group keys dynamically and securely.
- **Bike Connection Box:** This device aggregates data from various sensors and securely links a vehicle to the project platform, ensuring data integrity and confidentiality.
- **DLT-Based Data Tracing and Authorisation Engine:** Leveraging Distributed Ledger Technology (DLT), this engine provides a transparent and tamper-proof method for data tracing and authorisation, enhancing trust among stakeholders.
- **End-to-End Data Encryption:** This technology ensures that data is encrypted both in storage and during transit, protecting it from interception and unauthorised access.
- **Intrusion Detection System for Vehicles:** This system monitors vehicle networks for suspicious activities, providing early detection and response to potential security threats.

By implementing these technologies, the project aims to mitigate the identified threats and establish a secure, reliable platform for data sharing in the logistics and delivery sector.

Regarding the **visual interfaces**, this report primarily addresses the so-called "Virtual Cockpit", a virtual reality (VR) application developed to enable immersive and secure interaction with smart vehicles through TRACE centralised logistics platform. The "Virtual Cockpit" integrates key visual components designed for real-time control and monitoring, providing operators with a cohesive and user-friendly interface within a VR environment. Specifically, this report details the core application components, the structured information streams for input and output, and the security measures in place to safeguard data transmission.

The Virtual Cockpit application is designed to support three primary information streams, each designed to enhance interaction with smart vehicles within a VR environment:

- **Live Video Feeds:** Provides real-time visual input from the vehicles, enabling situational awareness and immediate responsiveness to operational changes.
- **GPS Data for Map-Based Tracking:** Allows continuous, accurate tracking of vehicle locations, displayed on an interactive VR map interface for precise navigation and coordination.
- **Predefined Emergency Signals:** Facilitates secure and efficient command transmission, with a limited set of standardised instructions (e.g., "return to base") to simplify communication and minimise risk.

Each information stream is secured through advanced privacy and data protection protocols, ensuring compliance with industry standards for safe, reliable data exchange with smart vehicles. By implementing the Virtual Cockpit, TRACE aims to offer an immersive and interactive way of monitoring smart vehicles in the next-generation logistics, as well as remotely intervening when needed.

This deliverable presents the activities of Task 4.6 – Cyber-Security and Data Protection and Task 4.7 – VR Interfaces for real time monitoring. As a follow-up, deliverable D4.6 will further report on details from the implementation, adaptations and testing results.

1.1 Scope of deliverable

Security and privacy. This deliverable first presents a brief analysis of the most relevant threats and attack vectors in the context of TRACE. Then, a high-level specification of security controls is given that aims to mitigate the identified threats. Finally, the subset of the identified controls to be developed and

demonstrated in TRACE is indicated and for each, a concise specification is given, along with the main integration and security considerations.

Out of scope for this report and foreseen for the deliverable D4.6 is the reporting on the exact implementation choices made for the security controls to be demonstrated, as well as the reporting on their integration in the respective demonstrators.

Visual interfaces. This deliverable reports the foundational design and security specifications for the visual interface components of the TRACE platform. It outlines the interface elements crafted to streamline user interactions, facilitate real-time data visualisation, and enhance decision-making within logistics operations. Key interface components include dashboards, data input forms, and control panels, each tailored to provide users with intuitive access to logistics insights. In addition, the deliverable emphasises security measures built into the interface, ensuring secure, encrypted communication between the platform, connected smart vehicles, and other integrated systems. This dual focus on usability and secure data handling is central to supporting reliable and efficient logistics operations in complex, data-driven environments.

1.2 Relation with other work packages/deliverables

The integration of the components under development in Task 4.6 is discussed in deliverable D3.1 “Report on reference architecture (A)”. The threat analysis and the security controls presented hereafter are elaborated for the TRACE architecture described in the same deliverable. The components reported in Sections 3.3 and 3.4 of the present deliverable are developed in Tasks 4.1 and 4.5, respectively; this deliverable only focuses on their security and privacy considerations.

Subsequently, the visual interface components and the underlying communication nodes reported in Section 3 are developed in the context of Task 4.7 - VR Interfaces for Real Time Monitoring and are discussed in D3.1 “Report on reference architecture (A)” as well.

Finally, the deliverable D4.6, a follow-up of the present deliverable, will report on details from the implementation, adaptations and testing results

1.3 Intended audience

The intended audience of this report includes the TRACE Consortium, as well as the general audience in the logistics sector, as well as providers of technologies for the said sector. Moreover, the report's threat analysis may be of particular interest to regulators.

1.4 Deliverable structure

The content is split into two sections:

- Sections 2 and 3 are dedicated to security and privacy activities performed by Task 4.6;
- Section 4 is dedicated to the visual interfaces activities carried out under Task 4.7.

In Section 2.1, a threat analysis of TRACE platform and use case identifies the assets, relevant threat events and potential vulnerabilities of TRACE. In Section 2.2, security controls are identified to address these vulnerabilities. Finally, in Section 3, the concrete security and privacy technologies developed by TRACE partners are specified in more detail and their security is discussed.

The deliverable continues with the analysis of the visual interfaces of TRACE (Section 3). Section 3.1 presents the components of the Virtual Cockpit, reporting their key features concerning the TRACE platform. Subsequently, Section 3.2 consists of an analysis of the input and output information streams that allow the communication between the visual interfaces and the vehicles, while finally, Section 3.3 reports a security and privacy analysis regarding the information included in the aforementioned streams, as well as the application environment of the Virtual Cockpit.

2 TRACE Security and Privacy Framework and Implementation Aspects

This section is dedicated to the security and privacy aspects of TRACE, addressed by Task 4.6.

2.1 TRACE Threat analysis

This section gives a brief analysis to identify the most relevant threats in the underlying scenario of TRACE and the main attack vectors that can allow these threats to be materialized. The methodology follows the NIST SP 800 30. This analysis covers controls that are either technical, or procedural, but fully relate to the TRACE technological platform. Procedural and other controls, which relate to the organization, that would adopt TRACE in the future are out of scope.

The analysis uses a semi-quantitative approach, assigning levels

- Very Low (VL)
- Low (L)
- Medium (M)
- High (H)
- Very High (VH)

to express sensitivity of assets with respect to security objectives and to characteristics of threat sources.

Note: In the context of this analysis, the term “sensitive information” refers to data that must be protected from unauthorized access to safeguard the privacy or security of an individual or organization.

2.1.1 System characterization

The objective of TRACE is to facilitate and optimize the logistics sector, by implementing a smart, data-driven system that optimizes the *global* resources across various logistics providers, integrates multimodal operations and backs everything up by automated digital contracts.

The approach of TRACE is to use real-time operational data in conjunction with various forms of machine learning algorithms to effectively detect disruptions, forecast the logistics workflows requirements (resources and routes), and assist in resource planning and route rescheduling tasks, allowing industrial stakeholders to become more proactive and account for complex factors.

The sought benefit is a **reduction of routes, lower operational costs and increase in the fuel efficiency of the vehicles.**

TRACE will further use blockchain operations to implement **shipments' traceability and financial operations in real time** and to handle financial management underlying synchromodal operations sharing the capacities of different logistics companies.

Stakeholders

The stakeholders that interact with TRACE and with one another are:

- **Customers**, that need to ship raw materials or goods through a supply chain network.
- **Carriers** that provide transportation services by transporting goods.
- **Logistics service providers** who provide logistics as a service at two different levels of integration:
 - **3PLs**, who offer order fulfilment, shipping, warehousing etc. as a service, taking care of the integral parts, or the entire supply chain management of the customer.
 - **4PLs**, who work as an intermediate between multiple 3PLs and customers, abstracting away even more of the supply chain management details and allowing to E.g., scale the distribution and warehousing network up or down more rapidly, increasing efficiency.
- **Infrastructure providers**, such as port authorities or intermodal hub managers.

TRACE architecture

The following Figure presents the high-level architecture of the proposed platform:

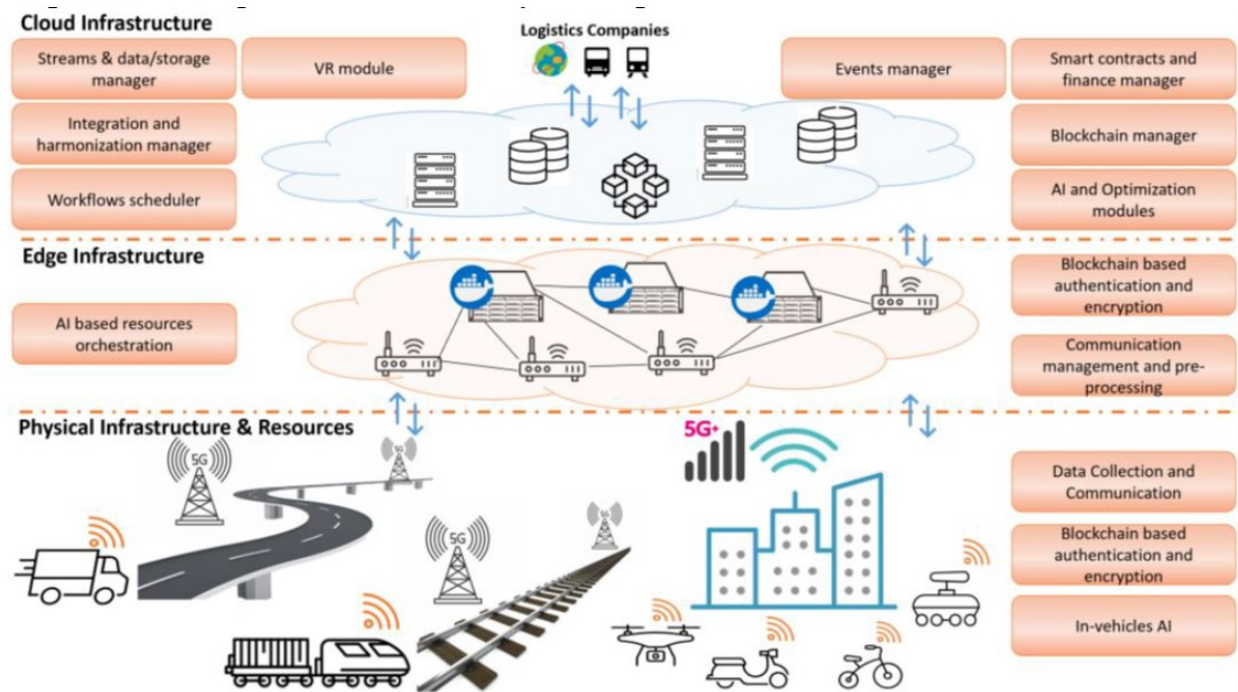


Figure 12-2 TRACE reference architecture.

The TRACE reference architecture, as specified by the project proposal (Figure 12-2) comprises the following key elements. The architecture has been further developed (see Figure 52-6), refer to the project deliverable “D3.1 Report on reference architecture (A)” for details.

1. **Physical infrastructure and resources:** TRACE will rely on cellular and medium-range wireless communication (4G/5G, IEEE 802.11) to implement most of the data collection from vehicles and infrastructure (or from Vehicle to Vehicle via infrastructure), with additional, ad-hoc networking infrastructure deployed along identified transportation corridors, terminals, and hubs.
In addition, short-range communication will be used between the vehicles and/or logistics infrastructure to ensure point-to-point/ device-to-device communication under all conditions, such as for coordination of vehicle platoons.
2. **Cloud backend infrastructure:** TRACE makes use of the cloud computing paradigm to implement a scalable platform that will be able to scale, ingest and process data from a number of logistics companies, coming directly from sensors through Data Sink components or through an interoperability layer from the companies’ own storage, using StreamHandler (based on the publish-subscribe paradigm).
3. **AI/ML scheduling/optimisation modules:** The scheduling and optimisation of (cross-stakeholder) shipment routes rely on machine learning approaches.

4. **Blockchain infrastructure:** The integrity and sequentiality of all TRACE transactions and data will be ensured using blockchain infrastructure, relying on the Algorand network.
5. **Integration and harmonisation module:** will ensure data from heterogeneous stakeholder databases is transformed and interpreted correctly inside the TRACE platform.
6. **Events management module:** TRACE platform will use the ingested data to detect events that may impact the effectiveness of the (overall) shipment delivery and consequently trigger corrective actions.
7. **Security and privacy protocols:** are in part the default protocols of the underlying telecommunications technologies and in part addressed at solution design and development phases. They are studied in this document.
8. **User interfaces:** will allow stakeholders, administrators and other parties to interact with TRACE platform.

Connection flow diagram

Figure 72-8 summarizes the TRACE architecture in a security-oriented connection-flow diagram; rather than an architecture diagram, a connection flow diagram does not capture a consistent level of abstraction but focuses on boundaries and directions, in which connections are made (see Figure 32-4). The boundaries highlighted by the diagram are what potential attackers must overcome and the existing connections can facilitate boundary traversal.

There are three main security environments:

1. The vehicle deployment environment, where the end-devices are potentially exposed to both remote and physical attacks. Especially, communication security vulnerabilities and threats are associated with the fact that communication between the stakeholder infrastructure and the TRACE platform can be performed over public networks (i.e. public mobile networks and the Internet).
2. The stakeholder infrastructure, which is less exposed to physical attacks but can, on the other hand, be an entry point for attackers, as the stakeholders can be expected to have heterogeneous security postures.
3. TRACE platform, with multiple entry points, such as the data upload through the StreamHandler, the interoperability layer, and the user-APIs.

In addition, TRACE will comprise blockchain-powered identity module, described in detail in the deliverable D4.3 Synchronodal operations and optimization of shared resources (A), with a security discussion in Section 3.4. The blockchain selected as the backbone for this module, the Algorand¹ network, is modelled as a security assumption in the present analysis. Nevertheless, potential vulnerabilities associated to this assumption are covered.

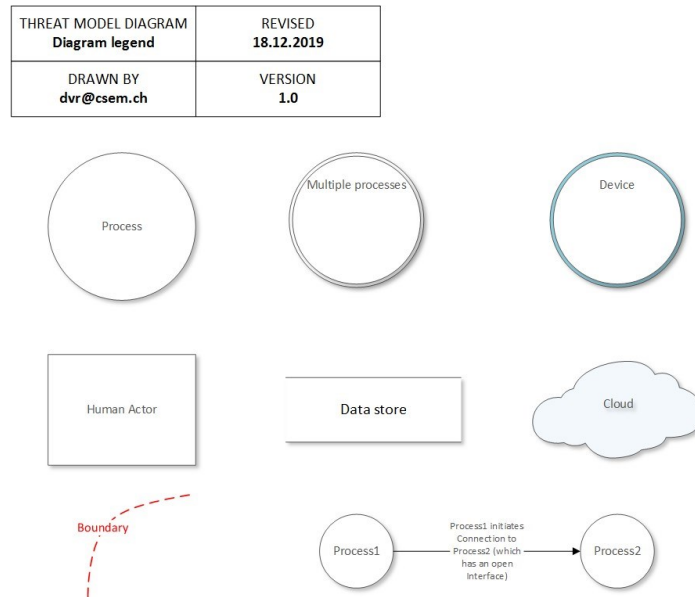


Figure 32-4 Interpretation of the symbols used in connection flow diagram.

¹ <https://algorand.co/>

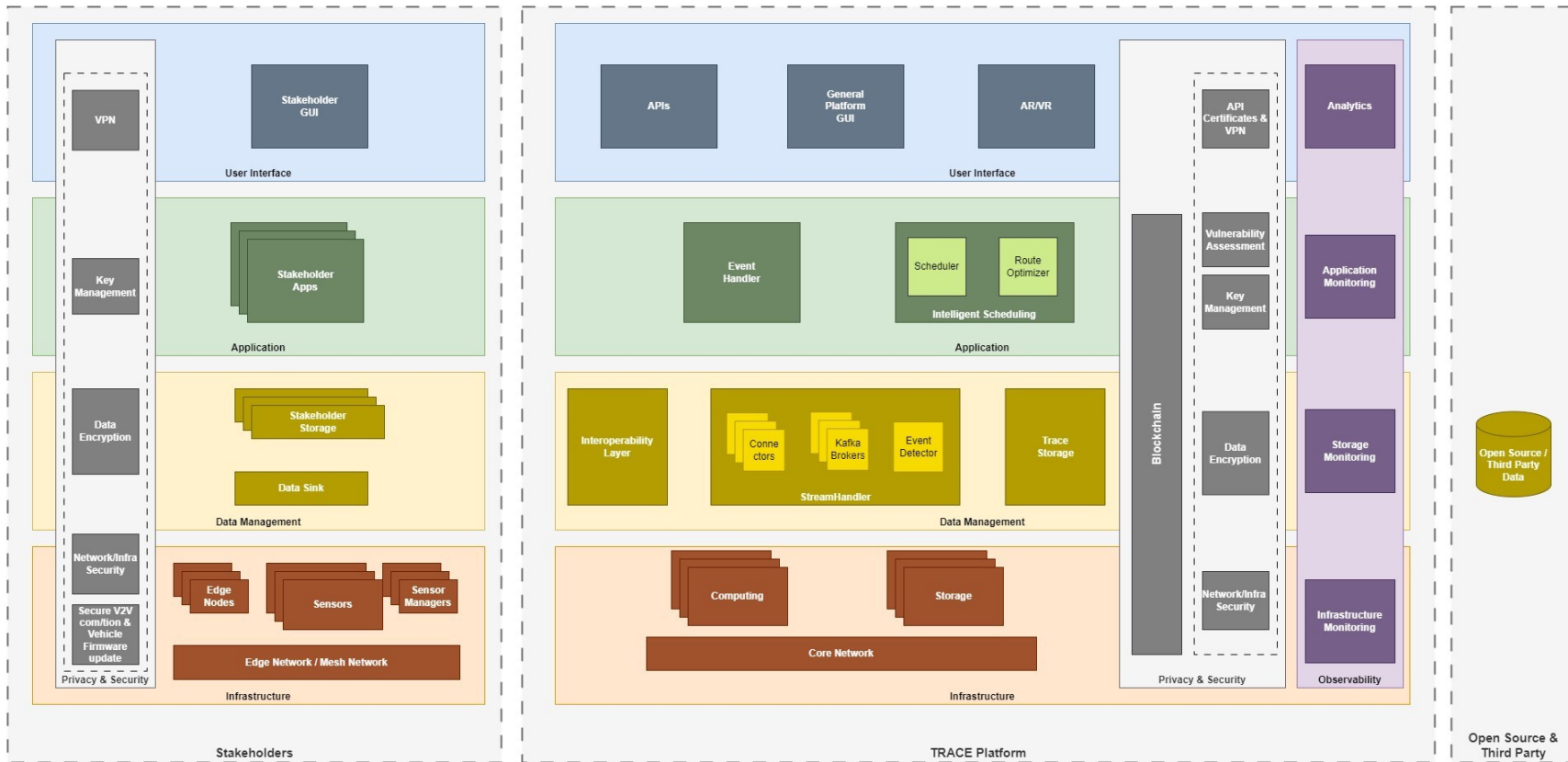
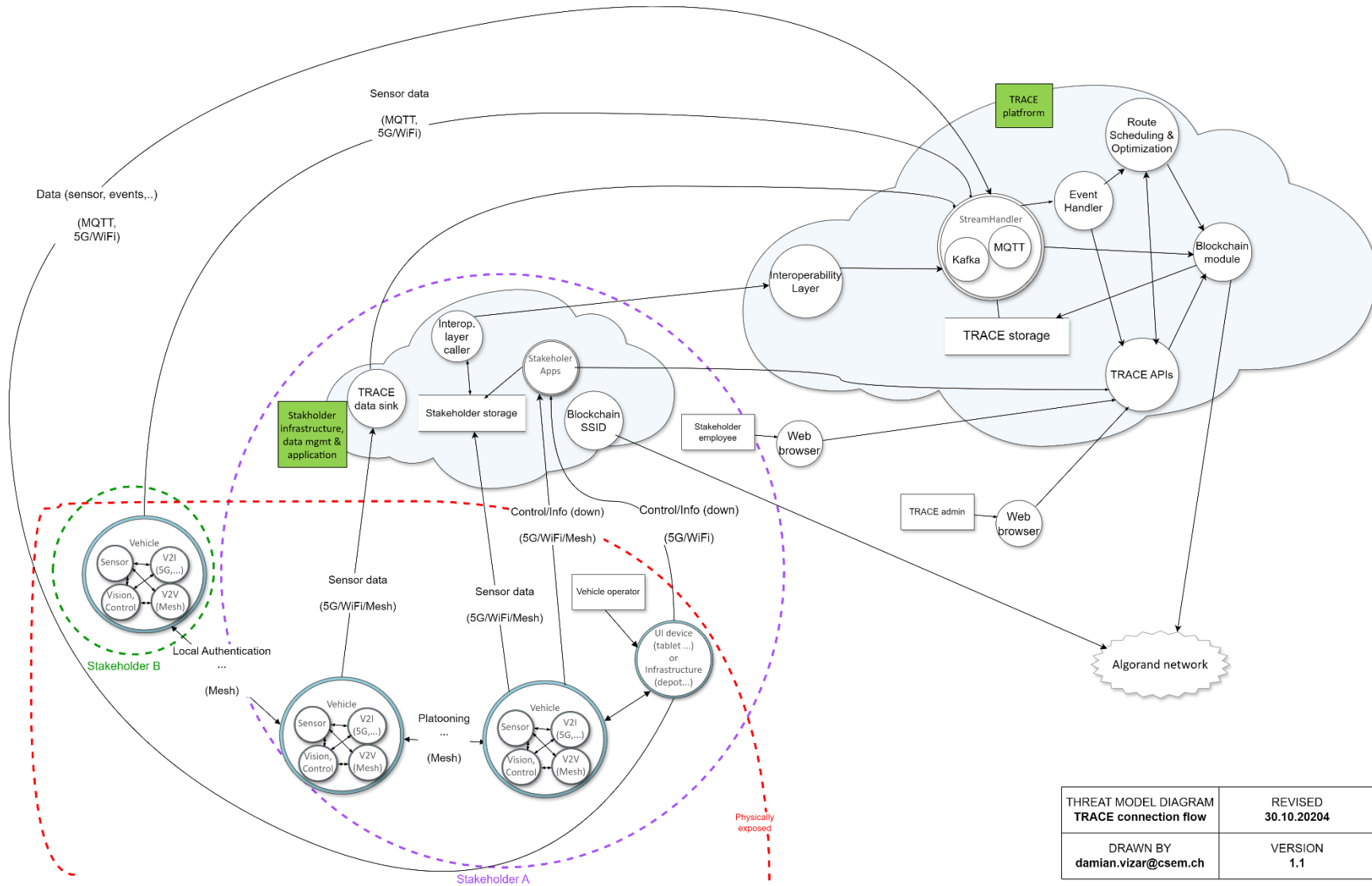


Figure 52-6 TRACE architecture



THREAT MODEL DIAGRAM TRACE connection flow	REVISED 30.10.2024
DRAWN BY damian.vizar@csem.ch	VERSION 1.1

Figure 72-8 TRACE connection flow diagram for threat modelling.

Scope

The present threat analysis covers the TRACE platform and its immediate dependencies: the sensors, V2I communication, as well as V2V communication and platooning (as this may have a direct impact on shipment security).

Assets

Table 1 provides a listing of TRACE, including the description of the asset, as well as security objectives for the asset, expressed as Is for Confidentiality (C), Integrity (I) and Authenticity (A), where the protection level is defined as the possible impact of the objective in question being violated for the given asset.

Table 1 TRACE assets and their security objectives

ID	Asset	Description	C	I	A	Notes
A1	Vehicle data	The data captured by on-vehicle sensors or the result of preprocessing of the latter, transferred to TRACE or to stakeholders own storage. Also includes event data, such as engine start/stop or parcel delivered.	M	H	H	The vehicle data may be considered sensitive by logistics companies, in particular w.r.t. their potential competitors that also use the services of TRACE. The integrity and availability of this data is critical, as the optimisations computed by TRACE depend on it.

A2	Vehicle control signals	For autonomous or platooned vehicles, the signals communicated to the vehicle allowing it to navigate autonomously.	L	VH	H	<p>The data for autonomous / platooned vehicle navigation should not be leaked to third parties. Yet, the impact if such a leak occurs is low, as it only pertains to the operations of the platoon in a given moment.</p> <p>The integrity of this data is paramount; if violated, this can be abused to hijack shipment or cause harm and economical damage.</p> <p>The availability of this data is slightly less critical, assuming a sane auto-stop in the absence of navigational control.</p>
A3	Shipment data	Information about the sender (the customer), the contents etc.	H	H	H	The more sensitive information associated to a shipment.
A4	Shipment metadata	Information about the destination, delivery constraints (deliver by), necessary for delivery optimisation	L	H	H	This data is not sensitive w.r.t. confidentiality, as it is necessarily shared (to some extent) inside the TRACE platform, to be able to deliver the shipment in the requested conditions.
A5	Stakeholder information	Stored in TRACE storage	M	H	H	Details about the stakeholder in the TRACE platform, including the identity, billing information etc.
A6	Events	As detected, created and processed by the Event Handler	L	H	VH	The events are not too confidential but require stringent integrity protection and must be available, in order to allow for a quick reaction and optimisations.

A7	Shipment scheduling data	The route and vehicles to be used for a given shipment	L	H	H	The scheduling of a shipment is not too sensitive, except for 3rd party access that could be abused to track potentially sensitive shipments for theft, for example. Protecting the integrity and availability is important to meet the expected optimization targets
A8	Allocation data	Information about the allocation of (transportation) resources across the TRACE participating stakeholders	L	H	H	Similar as above.
A9	Transactions	A record of all transactions traced by TRACE, E.g., enrolment of vehicles, enrolment of shipment, shipment events / cross-stakeholder transfers, delivery, etc.	M	VH	VH	Transactions may be moderately sensitive, as they reveal information on economical interactions. Their integrity and availability is critical both for traceability and auditability.
A10	Scheduling algorithms	All the algorithms used to create and optimize scheduling	M	VH	VH	These algorithms constitute the core USP of TRACE. If modified or unavailable, the platform will be critically disrupted.
A11	Self-drive and platooning-related IP	Signal processing and control algorithms enabling self-driving and platooning.	M	VH	H	These algorithms constitute a significant differentiator for some of TRACE's stakeholders. If modified or unavailable, physical or economic harm may be caused by the vehicles.

2.1.2 Threat Sources

Table 2 lists threat sources (or else attacker profiles) who may have an incentive to target TRACE. The threat sources are characterised in terms of their technical capability, their intent to perform an attack (and remain undetected), and the likelihood they would target TRACE in particular.

Table 2 Threat/Attack sources for TRACE

ID	Threat source	In scope	Capability	Intent	Targeting	Notes
AS1	TRACE users (logistics companies)	yes	L	M	H	When technically not too difficult, TRACE users may be tempted to get business intelligence on the fellow users who are also their competitors. In case of an arbitration, the users may be incentivised to deny responsibility and hamper audit procedures by abusing weak links in TRACE's authentication, data integrity and non-repudiability mechanisms.
AS2	Script kids	yes	L	L	L	Random amateur hackers may target TRACE for amusement and glory. Although it is not likely that a single such hacker would target TRACE in particular, their numbers raise the expectation that such a random attack will occur.
AS3	White hat hacker/hackivist	yes	H	L	M	White hat hackers may target TRACE to demonstrate the deficiencies in its security/privacy, motivated by activism and/or glory. As a system that has a potential to disrupt supply chains (when attacked), TRACE has a moderate chance to attract attention of academics and other white hat hackers. The white hat only cares to a limited extent about carrying out a real intrusion that goes undetected.

ID	Threat source	In scope	Capability	Intent	Targeting	Notes
AS4	Black hat hacker/ computer criminal	yes	H	M	M	Black hat hackers may target the TRACE to collect business intelligence and other sensitive data about the shipments. TRACE could also be targeted to trace high-value items to organize physical theft. Finally, TRACE's computational infrastructure could be abused as platform for further attacks. The black hat will try to cover their tracks.
AS5	Competitor	yes	M	M	M	A competitor may target TRACE to either steal the know-how, or parasite on TRACE's own services.
AS6	Curious vendor/ service provider	no	M	M	VL	The HW/OS/SW vendors of the used components (mainly phone and cloud) may opportunistically browse interesting data. They have a direct opportunity, especially on the cloud. It is unlikely that the provider would target TRACE in particular.
AS7	APT/ National agency	no	H	H	VL	A national agency has virtually unlimited resources and technical capabilities. This threat source is out of scope for this assessment.
AS8	Insider	yes	M	H	M	A developer or an admin may sabotage TRACE for profit or out of grudge.

2.1.3 Threat events

Table 3 provides a list of high-level threat events, i.e., events which, if materialised, would negatively impact TRACE, its users, or other stakeholders. The threat events are characterised in terms of the possible threat source, relevance, assets affected by the threat event and brief exemplification.

Table 3 TRACE threat events

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T1	Disruption of normal operation	AS1, AS2, AS4, AS5, AS8	Expected	A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11	<p>TRACE is no longer able to fulfil the intended functionality with the expected performance level.</p> <p>This threat events covers scenarios, such as:</p> <ul style="list-style-type: none"> - relevant events are not detected on time by TRACE - false events are created through a malicious tampering - malicious interference causes the proposed schedules to decrease overall efficiency - schedule information sent to different stakeholders diverges (E.g., A and B receive a diverging indication of a shipment handover point) - malicious interference results in shipment constraints not being met (E.g., "deliver by" constraint for perishable goods) - shipments routes to incorrect addresses

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T2	Confidentiality of sensitive information stored or processed by TRACE is disrupted due to a breach, leak, or data loss	AS1, AS2, AS3, AS4, AS5, AS6, AS8	Anticipated	A3, A5, A7, A9	<p>Sensitive information about shipments or TRACE users (stakeholders) is accessed by unauthorized parties.</p> <ul style="list-style-type: none"> - a user accesses information about another user (competitor) or their shipments - a third party extracts information about shipments to locate valuable goods for a theft - a third party extracts information on users and shipments to get business intelligence on users - a third party extracts information on shipments to track the supply chain of one of the TRACE end-clients - Personally identifiable data (in the sense of GDPR) is leaked, e.g., through ransomware <p><i>Note:</i> The extracted PII may be abused for further illegal activity, e.g., for phishing.</p>

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T3	Losses related to intellectual property	AS4, AS5, AS8	Anticipated	A10, A11	<p>Attackers steal IP that underpins the USP of TRACE, such as unauthorized access to</p> <ul style="list-style-type: none"> - TRACE optimization and scheduling algorithms - The IP enabling autonomous vehicle navigation and vehicle platooning - Infrastructure-as-code scripts allowing for a TRACE-like cloud architecture to be deployed w/o R&D investment

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T4	TRACE and its components are abused to cause harm or economical damage	AS4, AS7, AS8	Predicted	A1, A2, A4, A6, A7, A8, A10, A11	<p>Attackers abuse TRACE to cause damage, such as</p> <ul style="list-style-type: none"> - TRACE autonomous or platooning vehicles cause traffic incidents and/or injuries - TRACE scheduling pipeline is abused for a targeted supply chain disruption (of one of the end-clients) - TRACE scheduling pipeline is abused to cause a geographically targeted delivery disruption (to impact a significant proportion of deliveries made in a city, for example) - TRACE scheduling pipeline is abused to maximize disruption of traffic in a targeted location E.g., by concentrating delivery cars, by concentrating handovers to a hub or saturating flight corridors - TRACE suggests an unnecessary amount of cross-stakeholder transactions, ramping up the overall cost

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T5	Unsanctioned modification of function	AS1, AS3, AS8	Anticipated	A1, A4, A6, A7, A8, A9	<p>Users force TRACE to perform functions in an unintended way, or functions that were not intended at all, such as:</p> <ul style="list-style-type: none"> - Force the scheduling to favour a certain stakeholder over the others (E.g., give priority for shipment handovers when there is a lack of free delivery capacity) - Force the scheduling to favour a certain stakeholder as the provider of delivery capacity for other users, .i.e., schedule (significantly) increased handovers to this user - Generate smart electronic contracts with maliciously increased/decreased value

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T6	Abuse of resources	AS2, AS4, AS8	Anticipated	A1, A3, A4, A5, A6, A7, A8, A9	<p>The internet-facing infrastructure may be subject to resource-exhaustion (DoS) attacks. Cyber criminals may parasite on the resources (E.g., to mine Bitcoin), including by tampering with the infrastructure deployment code. Ransomware can be used by criminals to lock resources for extortion purposes.</p> <p>Abuse of resources could also significantly impact the operation cost of the platform by inducing transactions incurring costs but not generating revenue.</p>
T7	Delivery fraud	AS1, AS4, AS8	Predicted	A3, A4, A6, A7, A9	<p>Abusing TRACE to facilitate:</p> <ul style="list-style-type: none"> - a theft of shipments, E.g., by a third party that impersonates a TRACE user - a handover to an incorrect TRACE user, such that the receiving party is able to deny any responsibility - denial of a handover that actually took place by the receiving party

ID	Threat Event	Threat Source	Relevance	Affected Assets	Notes
T8	Infringement of data privacy regulations				<p>Adverse events include:</p> <ul style="list-style-type: none"> - Collection of personal data without proper authorization (informed consent by the data subject/owner) - Harmful effects on data owners in the data processing - Misuse of expertise in the data processing - Collection of additional personal data in the process - Transfer of personal data to third parties without proper authorization of the data owner - Misuse of personal data (e.g., processing that violates the declared intent) <p>These can be materialized for example as follows:</p> <ul style="list-style-type: none"> - Transmit parcel addressee information (name, address, further contact information) across TRACE users w/o data owner's knowledge and consent - Automated collection of personal data (e.g., web scraping, to train AI, route optimization etc.) - Accidental erasure of personal data

2.1.4 Vulnerabilities

Table 4 lists vulnerabilities, which, if present in the system, may allow an attacker materialise one or more threat events. The vulnerabilities are grouped by broad attack strategies, and the threats that can be enabled by each vulnerability are indicated.

Table 4 TRACE vulnerabilities

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V1.01	Poor authentication of the vehicle devices connecting to TRACE platform	Authentication	T1,T2,T4,T6,T7	
V1.02	Poor authentication of the vehicle devices connecting to stakeholder infrastructure	Authentication	T1,T2,T4,T6,T7	
V1.03	Poor end-to-end authentication of data ingested from vehicle sensors or infrastructure	Authentication	T1,T4,T6,T7	
V1.04	Poor end-to-end authentication of control signals and/or routing information sent to vehicles/personnel	Authentication	T1,T4,T6,T7	
V1.05	V2V comm. (mesh) has no or poor security (confidentiality, integrity, replay)	Communication	T1,T4,T6,T7	
V1.06	V2V comm. (mesh) is prone to man-in-the-middle attacks	Communication	T1,T4,T6,T7	
V1.07	V2V comm. (mesh) is not robust to long term device key leakage	Communication	T1,T4,T6,T7	
V1.08	V2I communication (over 4G/5G/Wi-Fi) has no or poor security (confidentiality, integrity, replay)	Communication	T1,T2,T4,T6,T7	
V1.09	V2I communication (over 4G/5G/Wi-Fi) is prone to man-in-the middle attacks	Communication	T1,T2,T4,T6,T7	
V1.10	V2I communication (over 4G/5G/Wi-Fi) is not robust to long-term device key leakage	Communication	T1,T2,T4,T6,T7	
V1.11	Poor end-to-end encryption of data ingested from vehicle sensors or infrastructure	Communication	T1,T2,T4,T6,T7	
V1.12	Debug port not effectively disabled (vehicle devices)	Physical attacks	T1,T3,T4,T6,T7	
V1.13	External memories used without encryption / authentication (vehicle devices)	Physical attacks	T1,T3,T4,T6,T7	E.g., a plain SD card used to store data or firmware

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V1.14	Poorly secured inter device/inter chip communication (vehicle devices)	Physical attacks	T1,T3,T4,T6,T7	
V1.15	Side channels (E.g., power consumption) leak information, E.g., on secret keys (vehicle devices)	Physical attacks	T1,T3,T4,T6,T7	
V1.16	Vehicle devices are prone to perturbation (E.g., voltage glitches)	Physical attacks	T1,T3,T4,T6,T7	E.g., to force unlock debug port
V1.17	Vehicle devices' UI implements no or poor authentication	Physical attacks	T1,T3,T4,T6,T7	E.g., some configurations are accessible without authentication
V1.18	Vehicle devices' UI does not correctly enforce authorization	Physical attacks	T1,T3,T4,T6,T7	E.g., administrative functions are available to any authenticated user
V1.19	The vehicle devices do not feature effective anti-tamper (or tamper-evident) features	Physical attacks	T1,T3,T4,T6,T7	
V1.20	Vehicle device firmware can be recovered, E.g., during an update	Software exploits	T1,T3,T4,T6,T7	
V1.21	Vehicle devices' firmware contains an exploitable vulnerability	Software exploits	T1,T3,T4,T6,T7	
V1.22	Vehicle devices reveal information upon exceptions and errors	Software exploits	T1,T3,T4,T6,T7	
V1.23	Vehicle devices' firmware cannot be updated remotely	Software exploits	T1,T3,T4,T6,T7	
V1.24	Exploitable logic issue in vehicle self-driving/platooning controls SW	Software exploits	T1,T3,T4,T6,T7	
V1.25	Well known vulnerable or misconfigured 3rd party SW components used in vehicle devices	Supply chain	T1,T3,T4,T6,T7	
V1.26	Vehicle devices' firmware can be replaced without an authentication check, or the check can be bypassed	Unsanctioned firmware replacement	T1,T3,T4,T6,T7	

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V1.27	Vehicle devices' firmware update package is lacking or has inadequate end-to-end encryption	Unsanctioned firmware replacement	T1,T3,T4,T6,T7	
V1.28	Private keys are reused across multiple vehicle devices	Weak key management	T1,T4,T6,T7	
V1.29	Vehicle device credentials cannot be effectively blacklisted	Weak key management	T1,T4,T6,T7	
V2.01	Data served through Interoperability layer is poorly authenticated	Authentication	T1,T4,T6,T7	
V2.02	Data served through Edge Data sink is poorly authenticated	Authentication	T1,T4,T6,T7	
V2.03	Edge data sink can push data to unforeseen topics	Authorization	T1,T4,T5,T6,T7, T8	
V2.04	Interoperability layer queries can access data that is not meant to be shared	Authorization	T1,T2,T4,T5,T6,T7, T8	
V2.05	Internet-crossing comm. (Edge and platform APIs) has no or poor security (confidentiality, integrity, replay)	Communication	T1,T2,T4,T5,T6,T7	
V2.06	Internet-crossing comm. (Edge and platform APIs) is prone to man-in-the-middle attacks	Communication	T1,T2,T4,T5,T6,T7	
V2.07	Internet-crossing comm. (Edge and platform APIs) is not robust to long term device key leakage	Communication	T1,T2,T4,T5,T6,T7	
V2.08	Data served through Interoperability layer can be altered by stakeholders without detection	Non-repudiation	T1,T4,T5,T7	
V2.09	Stakeholder Edge infrastructure poorly protected against physical attacks	Physical attacks	T1,T2,T4,T5,T6,T7	E.g., from access of a 3rd party to the machines in a server room
V2.10	TRACE Edge components contain exploitable vulnerabilities	Software exploits	T1,T2,T4,T5,T6,T7	
V2.11	Well known vulnerable or misconfigured 3rd party SW components used in Edge components	Supply chain	T1,T2,T4,T5,T6,T7	

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V2.12	Stakeholder SSID credentials are insufficiently secured	Weak key management	T1,T4,T6,T7	
V2.13	No or ineffective recovery is possible when stakeholder SSID credentials are compromised	Weak key management	T1,T4,T6,T7	
V3.01	TRACE platform session tokens entropy is weak	Authentication	T1,T2,T3,T4,T6,T7	
V3.02	TRACE platform sessions' lifetime is longer than necessary	Authentication	T1,T2,T3,T4,T6,T7	
V3.03	TRACE platform session tokens can be replayed	Authentication	T1,T2,T3,T4,T6,T7	
V3.04	There is no authentication of TRACE API calls	Authentication	T1,T2,T3,T4,T6,T7, T8	
V3.05	The web application design makes recovery of TRACE API keys possible	Authentication	T1,T2,T3,T4,T6,T7	E.g., native mobile or single-page applications
V3.06	No user authentication to the TRACE platform	Authentication	T1,T2,T3,T4,T6,T7, T8	
V3.07	The personal authentication method is vulnerable (E.g., to authorization code injection, cross-site request forgery,..)	Authentication	T1,T2,T3,T4,T6,T7, T8	
V3.08	There are backend resources that may be accessed without authentication	Authentication	T1,T2,T3,T4,T6,T7, T8	
V3.09	Insufficient access control of private endpoints	Authorization	T1,T2,T3,T4,T5,T6,T7, T8	
V3.10	Insufficient or permissive access control to key vaults	Authorization	T1,T2,T3,T4,T5,T6,T7, T8	
V3.11	Query authorization policy allows insecure direct object reference	Authorization	T1,T2,T3,T4,T5,T6,T7	

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V3.12	Sensitive folders allow browsing	Authorization	T1,T2,T3,T4,T5,T6,T7	
V3.13	Administration interfaces publicly accessible	Authorization	T1,T2,T3,T4,T5,T6,T7	
V3.14	Non-necessary resources (orphaned config files,...) hosted on server	Authorization	T1,T2,T3,T4,T5,T6,T7	
V3.15	IP/network restrictions insufficient	Authorization	T1,T2,T3,T4,T5,T6,T7	
V3.16	Business logic issue (unintended actions are possible)	Authorization	T1,T2,T3,T4,T5,T6,T7, T8	
V3.17	Stakeholders' private data accessible through queries made by other users w/o authorization	Authorization	T1,T2,T4,T5,T7	
V3.18	Backend resources (AI algorithms, databases) are vulnerable to statistical recovery attacks	Backend resource attacks	T2,T3	
V3.19	Insufficient logging of unusual analysis activity from users	Backend resource attacks	T2,T3, T8	
V3.20	No action is taken when a suspicious activity pattern is detected	Backend resource attacks	T2,T3, T8	
V3.21	Authenticity and integrity of input data not checked before processing a query	Backend resource attacks	T2,T3	
V3.22	TLS (or equivalent) not used	Communication	T1,T2,T3,T4,T5,T6,T7	
V3.23	TLS configuration weak (downgrade attacks, old versions...)	Communication	T1,T2,T3,T4,T5,T6,T7	
V3.24	Data sent out of the TLS tunnel	Communication	T1,T2,T3,T4,T5,T6,T7	
V3.25	TLS (or equivalent) not used for the internal cloud connections	Communication	T1,T2,T3,T4,T5,T6,T7	

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V3.26	Sensitive data (crypto. keys,...) transmitted in clear (other than authentication tokens)	Communication	T1,T2,T3,T4,T5,T6,T7	
V3.27	Cryptography used in the application will be defeated by quantum cryptanalysis	Communication	T1,T2,T3,T4,T5,T6,T7	
V3.28	Cloud exceptions and errors reveal detailed information	Configuration	T1,T2,T3,T4,T5,T6,T7	
V3.29	Query responses reveal detailed information	Configuration	T1,T2,T3,T4,T5,T6,T7	
V3.30	Admin/Developer prone to social engineering	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.31	Admin/Developer allowed to choose poor passwords	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.32	Admin/Developer number of incorrect password attempts not restricted	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.33	Non-production credentials reused in production system	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.34	2-factor authentication not enforced	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.35	2nd factor vulnerable	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.36	No notifications of suspicious logins	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.37	No procedure to react to suspicious login notifications	Identity theft	T1,T2,T3,T4,T5,T6,T7	
V3.38	External inputs (HTTP params, headers, cookies, URI etc., device data) are used with poor sanitization	Injection	T1,T2,T3,T4,T5,T6,T7	
V3.39	Insufficient protection from CSRF attacks	Injection	T1,T2,T3,T4,T5,T6,T7	

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V3.40	External inputs used insecurely in code queries (OS, DB, programs)	Injection	T1,T2,T3,T4,T5,T6,T7	
V3.41	External input sanitization can be bypassed	Injection	T1,T2,T3,T4,T5,T6,T7	
V3.42	External input is displayed on user or administrative pages (XSS vulnerability)	Injection	T1,T2,T3,T4,T5,T6,T7	
V3.43	Vulnerable configuration of service or components	Injection	T1,T2,T3,T4,T5,T6,T7	
V3.44	No or insufficient version control used	Insider	T1,T2,T3,T4,T5,T6,T7	
V3.45	Insufficient authentication of commit authors (in git)	Insider	T1,T2,T3,T4,T5,T6,T7	
V3.46	Developer access not logged	Insider	T1,T2,T3,T4,T5,T6,T7, T8	
V3.47	Developers can modify or erase activity logs	Insider	T1,T2,T3,T4,T5,T6,T7, T8	
V3.48	Code is not properly reviewed	Insider	T1,T2,T3,T4,T5,T6,T7	
V3.49	Admin access not logged	Insider	T1,T2,T3,T4,T5,T6,T7, T8	
V3.50	Admins can modify or erase activity logs	Insider	T1,T2,T3,T4,T5,T6,T7, T8	
V3.51	Algorand network security assumption not satisfied (no honest supermajority)	Non-repudiation	T1,T4,T5,T7	
V3.52	Algorand consensus protocol or implementation is flawed	Non-repudiation	T1,T4,T5,T7	
V3.53	TRACE smart contracts or their implementation are flawed	Non-repudiation	T1,T4,T5,T7	E.g., containing loopholes that can be exploited

ID	Vulnerability description	Linked attack strategies	Applicable to threat events	Comment
V3.54	TRACE smart contracts do not effectively link integrate the underlying data in a non-repudiable way	Non-repudiation	T1,T4,T5,T7	E.g., a handover of shipment with a deniable participation or shipment ID
V3.55	It is possible for a transaction (E.g., shipment handover) to be finalized in the TRACE platform without the transaction being finalized in the real world	Non-repudiation	T1,T4,T5,T7	E.g., convince the platform a shipment has been handed over without this actually happening, to incriminate another user
V3.56	It is possible to change the order in which transactions appear to have occurred	Non-repudiation	T1,T4,T5,T7	
V3.57	Lack of protection against subdomain takeover	Resource exhaustion	T1,T6	
V3.58	Insufficient limitation in user actions that incur cost but do not generate revenue	Resource exhaustion	T1,T6	
V3.59	The architecture does not offer sufficient protection from (D)DoS attacks	Resource exhaustion	T1,T6	
V3.60	There are user actions that cost but do not generate revenue	Resource exhaustion	T1,T6	
V3.61	Inadequate backup of TRACE data and configuration	Resource exhaustion	T1,T6	
V3.62	Poor protection of TRACE backup	Resource exhaustion	T1,T6	
V3.63	Vulnerable or misconfigured services are exposed	Supply chain	T1,T2,T3,T4,T5,T6,T7	
V3.64	Unpatched components or services present	Supply chain	T1,T2,T3,T4,T5,T6,T7	

2.2 Security controls for TRACE architecture

This section describes high-level security controls needed to harden TRACE against the attacks identified in Section 2.1.2. Not all these controls will be implemented in the project since e.g. some are subject to global standard specifications (i.e. 3GPP, IEEE, IETF, etc.), some are addressed by design principles, while other need to be addressed when TRACE is configured and deployed in commercial environments. Therefore, the list should rather be seen as guidelines for future adopters and as a basis for risk analyses of deployments.

Table 5 TRACE security controls

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C01	Devices connections to the TRACE platform use an appropriate authentication method.	V1.01, V1.02	E.g., using tokens or digital certificates.	
C02	All data uploaded to TRACE platform is end-to-end authenticated.	V1.03, V2.01, V2.02, V3.21	E.g., using digital signatures or authenticated encryption	
C03	Control signals, routing information and any other downstream communication is end-to-end authenticated.	V1.04	E.g., using digital signatures or authenticated encryption	
C04	V2V communication protocol includes a security layer for confidentiality, authenticity and anti-replay features	V1.05		SHOULD
C05	The V2V communication key management should resist to a certain fraction of devices being compromised and resist man-in-the-middle attacks if communication sessions are established dynamically	V1.06, V1.07		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C06	Use standard security protocols (EAP, IPSec, TLS etc.) at V2I network layer and TLS (or equivalent) for communication between the device application components and the TRACE platform, with an appropriate configuration	V1.08, V1.09, V1.10	The configuration shall prevent downgrade attacks etc.	SHOULD
C07	Data uploaded to TRACE platform is end-to-end encrypted	V1.11	E.g., using secure authenticated encryption	SHOULD
C08	Debug ports of all devices on the vehicles are disabled no later than after key provisioning AND prior to their installation,	V1.12		
C09	If external memories are used by the vehicle devices, the data stored on these shall be authenticated and should be encrypted.	V1.13	The encryption requirement may be relaxed if the device design ensures all sensitive data is stored in the internal memories.	SHOULD
C10	All communication between devices and/or MCUs on a single board shall ensure authenticity and replay protection and should ensure confidentiality	V1.14		
C11	The vehicle devices should employ cryptography implementations that resist to side channel attacks and/or perturbation attacks.	V1.15, V1.16	Either the HW platforms are certified/claim security against these attacks, or specialized SW is used instead.	SHOULD
C12	All UI functions on vehicle devices shall be subject to a secure user authentication.	V1.17	E.g., using a passcode with a suitable policy on timeouts between trials	SHOULD
C13	All administrative functions in the vehicle devices' UI shall only be available to a distinguished admin user	V1.18		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C14	The vehicle devices feature anti-tamper or tamper-evident features.	V1.19	A procedure is also needed to regularly inspect the devices for signs of tampering.	
C15	Vehicle device firmware update package is end-to-end encrypted	V1.20, V1.27		
C16	All software development (for vehicle devices, Edge and cloud) involves procedures and tools ensuring the quality of code and minimizing the chance of exploitable vulnerabilities.	V1.21, V1.24, V3.48	The procedures involve mandatory code reviews and the tools involve static code analysis and a CI/CD with mandatory test gates.	
C17	Production firmware shall reveal no debugging information, error strings, etc.	V1.22	E.g., by correctly using deployment environments	
C18	All vehicle devices implement a secure boot and firmware update over the air.	V1.23, V1.26		
C19	A SW BOM management procedure is in place, ensuring any vulnerability in a 3rd party component is identified and patched without undue delay	V1.25, V2.10, V2.11, V3.63, V3.64		
C20	The key management of TRACE should ensure the use of personalized keys for vehicle devices	V1.28		
C21	TRACE should implement vehicle device key revocation	V1.29		
C22	StreamHandler implements and effective least-privilege access control for all connections	V2.03		SHOULD

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C23	The TRACE Edge adapter for data access implements effective access control	V2.04		SHOULD
C24	Use TLS (or equivalent) for all Internet-crossing as well as internal communication, with an appropriate configuration	V2.05, V2.06, V2.07, V3.22, V3.23, V3.24, V3.25		MUST (Internet crossing), Else SHOULD
C25	Use an appropriate non-repudiation cryptographic primitive	V2.08	E.g., digital signatures or committing symmetric encryption	
C26	TRACE users (stakeholders) shall ensure (a minimal) physical isolation of Edge resources relied on by TRACE	V2.09		
C27	Best-practice security configurations are applied to 3rd party software and services	V2.11		SHOULD
C28	TRACE SSID key wallets shall employ key leakage mitigation measures	V2.12	E.g., HW wallets or threshold cryptography	
C29	An (identity) recovery procedure should be implemented for the case of an SSID takeover	V2.13	for pilots, a simple procedure with direct verification through alternative channel suffices	MUST
C30	Generate session tokens using a cryptographically strong randomness source or other source ensuring a sufficient entropy.	V3.01		
C31	Session expiry time is configured according to best practice	V3.02		
C32	Expired session tokens can be detected and cannot be reused	V3.03		
C33	API calls are authenticated, e.g., using API keys.	V3.04		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C34	Assuming a *non*-single page architecture, this vulnerability is not very likely. In a single-page architecture, it cannot be prevented.	V3.05		
C35	A secure login is implemented (this can be done using federated login from an identity provider)	V3.06		
C36	Implement PKCE OAuth2 flow or equivalent	V3.07		
C37	(1) A catalogue of all APIs shall be maintained (2) A regular review to ensure all (except a whitelist) are subject to personal authentication	V3.08		
C38	Sound access control of cloud private endpoints disallowing direct queries.	V3.09		SHOULD
C39	Cloud architecture shall enforce internal authentication and restrictive access control to access key vaults	V3.10		SHOULD
C40	Least privilege authorization configuration shall be put in place, such that authorization shall be checked for every object a user attempts to access.	V3.11, V3.12, V3.16		
C41	Administration interfaces shall not be accessible without authentication.	V3.13		MUST

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C42	The cloud architecture shall be documented, and regular reviews put in practice to ensure only necessary resources are hosted.	V3.14, V3.16		
C43	IP filtering of internal network shall be implemented to mitigate directory enumeration and other attacks	V3.15		
C44	Data at rest is encrypted, using appropriate encryption algorithm ensuring confidentiality and integrity. Where applicable, user-specific keys are used, such that authorized access granted does not provide the ability to access other data than intended.	V3.17		SHOULD
C45	The rate at which users can trigger AI inference is (progressively) limited	V3.18		
C46	User activity and accesses are logged and automatically evaluated and reported for unusual patterns	V3.19, V3.20	The minimal scope of the logging shall include user login, log out, the information about the source of the login (if applicable). The monitoring and automatic evaluation should be implemented in real-time.	SHOULD
C47	No keys shall be transmitted in clear text in API queries	V3.26		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C48	TRACE security architecture should allow for a timely transition to post-quantum secure cryptography	V3.27	The tie of transition should be based on the recommendations of national security agencies (such as ANSSI or NSA) and shall respect the data security period. It is recommended to use hybrid schemes combining conventional cryptography with post-quantum NIST standards.	
C49	Production platform instance shall disable and/or sanitize all error information	V3.28		
C50	No unnecessary information shall be contained in query responses or their metadata	V3.29		SHOULD
C51	Personnel with administrative privileges shall be sensibilized to cybersecurity and social engineering practices (e.g., by taking training).	V3.30		SHOULD
C52	Enforce a sound password policy following best practices. Alternatively, use passwordless login (e.g., FIDO2).	V3.31		SHOULD
C53	Login shall be temporarily blocked after a few incorrect attempts for a given user, with sufficient/exponentially increasing timeout.	V3.32		MUST
C54	Policy prohibiting the reuse of credentials and keys from non-production (development, staging) environment in production instances shall be put in place.	V3.33		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C55	Two-factor authentication shall be mandatory for all privileged users and developers. It shall be available for all users and should be mandatory.	V3.34		SHOULD
C56	Use a best practice to implement second authentication factor	V3.35	Use authenticator apps, FIDO2 or similar, avoid SMS.	
C57	Criteria for suspicious logins shall be defined AND suspicious logins trigger notifications/emails to admins.	V3.36		
C58	A procedure specifying exact steps to react to a suspicious login notification is defined AND a responsible person designated at all times	V3.37		
C59	A Web Application Firewall or equivalent control shall be implemented.	V3.38	If no native controls (such as a web application firewall) exist, then best practice input sanitization shall be applied.	SHOULD
C60	Anti-CSRF protection shall be used.	V3.39	Either native anti CSRF features of the application framework shall be used, or best practice guidelines followed (e.g., following OWASP's cheat sheets)	SHOULD
C61	Use best practices when handling external inputs (e.g., parametrized DB queries, general sanitization, per-language measures, WAF)	V3.40	This includes the handling of data coming in from sensor devices and through the interoperability module, as either of these sources might get compromised	SHOULD
C62	Cloud architecture shall not permit the input sanitization to be bypassed.	V3.41	This means there is only one path for external inputs/data to be ingested (per data type).	SHOULD

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C63	Best practice XSS defences shall be implemented (native defences and thorough input sanitization)	V3.42		SHOULD
C64	Best practice secure configurations shall be adopted for all external components or services.	V3.43		SHOULD
C65	Use a version control tool with appropriate configuration, e.g., a git-based solution using protected branches, pull requests with mandatory code reviews, strong user authentication etc.	V3.44		
C66	Developers shall be authenticated for each pushed commit (ensuring that the claimed identity of a commit matches the authenticated user)	V3.45	In some git flavours, it is possible to authenticate with credentials of user A and push commits pretending to come from B by default. In most, a possibility of a strong configuration exists.	
C67	Git server logs shall be configured and sufficiently verbose.	V3.46		
C68	Developers shall not have admin rights to manage the version control	V3.47		
C69	Ensure all admin activity is logged	V3.49		SHOULD
C70	Use cloud native features that generate immutable logs, if available	V3.50		
C71	Use abstraction tools to develop a chain-agnostic application, design a procedure allowing rehosting on another chain.	V3.51, V3.52		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C72	Apply formal verification tools when designing and/or implementing TRACE smart contracts.	V3.53		
C73	The design and implementation of smart contracts shall incorporate compulsory peer review process.	V3.53		
C74	All cryptographic primitives used when linking data to smart contracts shall have non-repudiation / committing properties	V3.54	E.g., when linking symmetrically encrypted data to a distributed ledger, the underlying encryption shall be committing	
C75	TRACE security architecture shall involve a local, cryptographic confirmation of any transaction	V3.55	E.g., when a shipment is handed between two logistic companies, the involved vehicles or personal devices shall run an interactive authenticated proof protocol over a local communication channel (e.g., BLE), such that the result of this is incorporated in the corresponding smart contract.	
C76	Use a hash chain (or equivalent) to strongly sequentialise all transactions.	V3.56	If data or event aggregation is used to link multiple entries to a single block, the aggregation technique shall also ensure sequentiality.	
C77	Automate the renewal of (sub)domain lease	V3.57		
C78	Impose limits on queries that induce cost (e.g., account creation)	V3.58		
C79	Use cloud native protection features.	V3.59		
C80	Directly monetizes the most computationally intensive actions	V3.60		

ID	Security control	Mitigated Vulnerabilities	Notes	Recommended in demonstrators
C81	Effective backup procedures are put in place, with a sufficient backup frequency	V3.61		
C82	The backup data are well isolated (in terms of authentication and/or authorization of administrative access)	V3.62		
C83	Data related to operation of vehicles is being logged continuously and automatic detection and/or classification of unusual patterns is put in place to detect possible intrusions.	V16, V21, V22, V24, V25		

2.2.1 Controls compulsory for demonstrators

The TRACE development and demonstration platform will not implement all the security features identified in the previous section, this not being the objective of the project, nor being compatible with the resources available. Nevertheless, a subset of the identified security controls is identified in the column “**Recommended in demonstrators**” of Table 5, which are recommended for inclusion in the demonstrators, so as to ensure a level of security that will prevent a disruption of the platform and mitigate the risks to the data used in the pilots. These minimal security controls are marked with “MUST” or “SHOULD”, in the sense defined by IETF RFC Style Guide [1].

The method to elaborate this list consists in the partners in Task 4.6 proposing the initial list of compulsory controls presented in this deliverable and consequently, a collaborative process with other technical tasks and pilots (hence the prevalence of “SHOULD”) will yield the final configuration of controls, which will be documented in the follow-up deliverable D4.6. Note that this is a voluntary initiative of the TRACE consortium.

3 Security controls developed in TRACE

This section describes in more detail a subset of the security features identified in Section 2.2 that will be implemented and demonstrated in the project TRACE. Each of these controls is covered in a dedicated subsection.

3.1 Secure boot and firmware update for V2V communication module

Component code: V2VcomBOOT

Related to control IDs: C15, C18

The V2V communication module of TRACE is a constrained embedded device (based on a STM32 system on chip family) that is connected with other onboard systems of the vehicles with SPI, UART (or similar) and enables wireless, local communication between vehicles without the need for any infrastructure. This communication will be primarily used for synchronization and exchange of information when vehicles are platooning, such as in e-bike platoons with a single driver. The V2V communication module might also be used to interwork with a V2I communication module that will enable communication between the vehicle and the platform when cellular connection is available. For more information on the V2V communication module specifications, please refer to the TRACE deliverable “D4.1 Infrastructural Elements of the TRACE Platform (A)”.

The V2V module may be exposed to physical attackers (e.g. when installed on bikes, which may be left unattended when the driver delivers a package in a building) as well as to remote attackers (over the wireless communication interface). If these attackers manage to permanently compromise the firmware, this can lead to material losses and injury in the worst case (when the changes are used to actively sabotage the platooning) or as an entry point to other onboard systems.

This component will implement **secure boot and firmware update over the air (FUOTA) for the V2V communication module**. Secure boot will ensure that only intended, unmodified firmware will run upon every reset, significantly reducing the space of possible attack vectors. Secure FUOTA, on the other hand, will allow the firmware to be updated remotely, allowing patching of vulnerabilities at scale. In particular, the component will support a mass-FUOTA, to allow SMEs maintain a fleet of vehicles effectively.

3.1.1 Specification

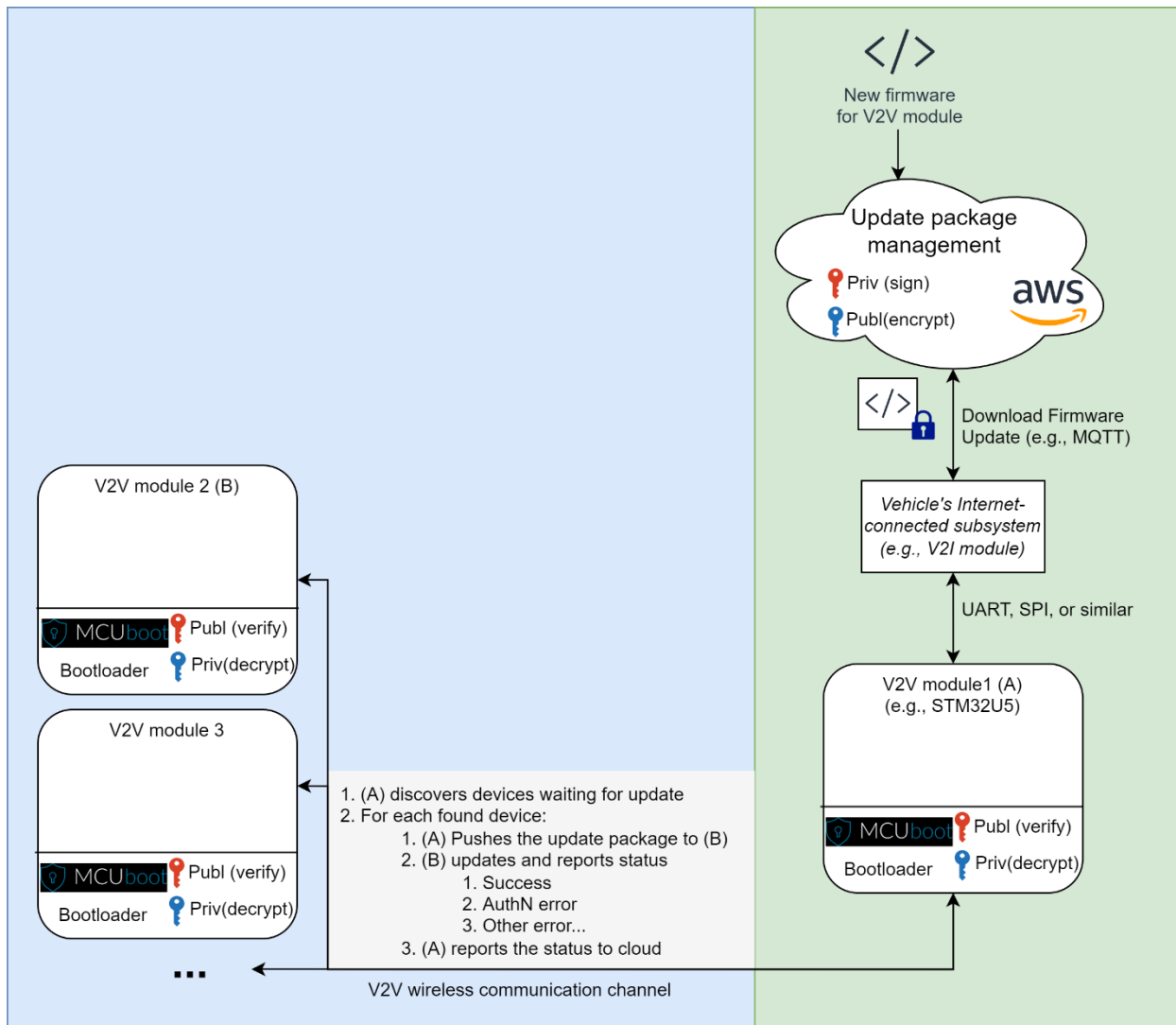


Figure 93-10 Secure boot and FUOTA from V2V communication module. The component will be developed in two iterations. In the first iteration (green), an automatic FUOTA for vehicles with direct Internet connection (e.g., 5G) is implemented. In the second iteration, the support for an automatic propagation of the update to vehicles that may not have Internet connection (e.g., for a more cost-effective fleet with platooning) is added (blue).

The embedded implementation of secure boot and FUOTA will be based on the industry-tested MCUBoot framework [2], with the following key features:

1. Firmware binary is signed with a private key managed by update package management backend.
2. Prior to execution, firmware integrity is verified with associated public key upon each reset. If a problem is detected, device halts.

3. Firmware update packages contain signed firmware binary, which can be additionally encrypted using device-specific public keys.
4. The device decrypts the package and verifies it upon every attempted update.

The firmware update package will be delivered from the backend to the V2V module using a wireless communication interface. In the first development iteration, the delivery will be implemented with help of the Internet connection of another onboard component, such as the V2I module. In the second iteration, support of offline vehicles will be implemented, whereby the V2V module of an Internet-connected vehicle will automatically propagate the update to other nearby vehicles using the V2V wireless interface itself. The key consideration here is that this communication will not require any secure device pairing to work, in order to enable an autonomous mass update; the protection from abuse will be ensured by the FUOTA security itself.

The backend used to securely generate (sign and encrypt) update packages, as well as to manage the fleet of V2V modules and keep track of the status of their updates will use AWS native cloud toolbox.

3.1.2 Security considerations

This component has the following security objectives. In this section, “firmware” refers to the V2V module firmware.

Objective	Attack vectors mitigated	Attack vectors non-mitigated	Remarks
Firmware integrity	Modification of the firmware stored in the internal non-volatile memory of the V2V microcontroller; these modifications are discovered at reset and firmware is not executed.	SW exploits that inject malicious code in RAM or in the parts of internal non-volatile memory that is not intended for firmware (e.g., data storage), without making the device crash. The firmware is not protected from modification as such. There is no guarantee against perturbation (e.g., voltage glitching) attacks.	To protect from physical attacks, debug ports must be disabled. Memory protection of bootloader from write is required.

<p>Firmware package integrity</p>	<p>Attempt to push a modified, or completely fabricated firmware update package; the attempt is rejected based on an invalid signature, or an inferior version number.</p> <p>Attempts to brick the device by pushing a corrupted firmware; integrity check is performed prior to an installation attempt.</p>	<p>Resource exhaustion attacks that repeatedly invoke an update to drain computational resources and/or battery are not prevented.</p> <p>Security against perturbation attacks is not guaranteed.</p>	
<p>Firmware package confidentiality</p>	<p>Attempts to extract the binary and intellectual property therein; thanks to appropriate encryption.</p> <p>Extraction of private keys from a device (to have perpetual access to updates); mitigated, provided decryption keys are personalized.</p>	<p>Extraction of firmware from the internal memory of the V2V module; there firmware is not encrypted, as it needs to be executed.</p>	<p>For the mitigation of key extraction attacks, it is also necessary to detect the extraction and have an effective revocation in place (so no update packages are issued for non-whitelisted devices, for example). This is out of scope for this component.</p>

3.1.3 Integration considerations

Integration of this component comprises the following:

- **V2V module bootloader.** The V2V module must adopt the secure bootloader responsible for secure booting and applying the updates. The bootloader must be protected from writes and

reads by the application firmware (to protect the private decryption key), with help of the MCU's memory protection. The bootloader is independent from the application firmware.

- **Mass update functionality in V2V module application.** The application firmware must implement the communication elements needed for the module to be found by another V2V module, receive an update and report back the result after an attempted update. This may imply certain elements of fleet management (unique identity and authentication credentials). Finally, the application must write the received update package into an update slot that is also recognized by the bootloader.
- **Internet-connected vehicle module.** The V2V module needs to communicate with an Internet-connected module on the vehicle, either over UART, SPI or a similar device to device communication technology. The latter is needed to bridge the communication to the update package management in the cloud.
- **Update package management.** The logic and communication elements allowing the creation and distribution of update packages, as well as certain fleet management elements (e.g., management of personalized firmware encryption keys and device identities) must be integrated in the cloud platform. It is recommended that the firmware signing key be stored and handled in a native secure key management block.

3.2 V2V communication security

Component code: V2VcomSEC

Related to control IDs: C04, C05

The local wireless communication between vehicles (through the V2V communication modules, please refer to the deliverable “D4.1 Infrastructural Elements of the TRACE Platform (A)” for details on the protocol) will allow synchronization and exchange of information when vehicles are platooning, and possibly support other use cases when cellular connection is not available. As such the communication needs protection of:

- **Integrity:** else platooning function may be disrupted, resulting in financial damage or even harm.
- **Against replays:** same as above, or platooning vehicles may be hijacked.
- **Confidentiality:** to prevent tracking and protect sensitive information (e.g., if the connection is used to transmit information about the cargo, for example).

These properties require an implementation of a secure channel, which is adapted to a constrained communication channel, such as the ones previously designed by CSEM in projects AMPWISE [3] and OffshoreMuster [4], for example. The challenge for TRACE V2V communication is the key management. On one hand, the platooning vehicles need to form ad-hoc groups, where it is desirable for all vehicles in the group to be privy to the communication (especially for a low-latency communication with the vehicle leading the platoon). On the other hand, the V2V module is exposed to both physical and remote attacks, as discussed in Section 3.1. The simple solution of letting all V2V modules share the same secret communication key is thus too vulnerable to key extraction; a single successful attack could compromise the security of all vehicles in the fleet.

To decrease the risk due to individual V2V module attacks while enabling secure communication in ad-hoc groups, **secure key establishment based on threshold cryptography for V2V communication module** is developed. Threshold cryptography helps reduce risks by distributing keys among multiple parties. To perform the cryptographic operation (decrypt, sign a message, etc.) requiring the shared-out key, it requires a minimum number of participants (a threshold) to collaborate. This approach enhances security and fault tolerance, as no single party can compromise the system.

3.2.1 Specification

The security of V2V communication consists of two components:

1. A protocol for an ad-hoc group session key distribution.
2. A security layer for secure communication in a group.

Ad-hoc group session key distribution protocol

For this protocol, each V2V module must be provisioned with

- An asymmetric key pair, with the public key enclosed in a digital certificate signed by an application-specific root private key. The digital certificate may have a custom optimized format.
- A distributed pseudorandom function (DPRF) [5] key share.

A DPRF uses a secret key that is split into share using a cryptographic secret sharing [6], such that a threshold T of key share owners must interact in order to apply the DPRF. The threshold T is a parameter, the lower, the less over head the DPRF protocol incurs, the higher, the more difficult it is to break the security (as any number $< T$ of key shares compromised is useless for an attacker). Together, these T participants run an interactive protocol to evaluate

$$Y = \text{DPRF}(K, X)$$

To obtain a pseudorandom value Y for a public input message X . The value Y is computationally indistinguishable from a uniformly distributed bit string and may be used as a group key, when kept secret. The use of a DPRF requires point-to-point secure communication between the participants of the protocol. This will be ensured using the asymmetric key pair and certificates.

In the key distribution protocol (Figure 113-12, the group setup initiator will start the protocol by soliciting nonces from the would-be group participants. Then the nonces will be combined in a message X , and then DPRF will be run by all group participants on X , using the output Y as the group key. A simple key confirmation (using authenticated encryption) will be performed. Crucially, a group participant will only accept the group key if it can see its nonce contribution in the message X , ensuring freshness from the perspective of every participant. The DPRF implementation will be optimized, as the runs of the DPRF by the various group members feature (almost) exactly the same messages.

The recommendation is to use $T \geq 2$, such that a simple exchange (based on a fair coin flip [7]) may be used to establish key for two V2V modules (a “group” of 2).

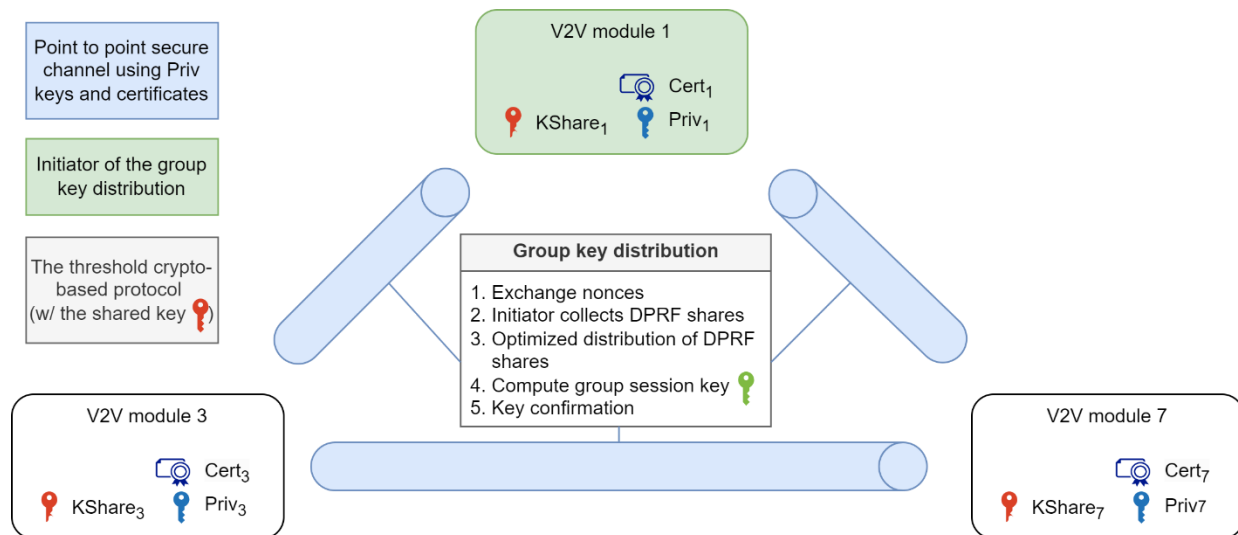


Figure 113-12 Ad-hoc group key distribution for V2V communication in TRACE. Here an example with threshold $T=3$, with V2V modules identified with IDs “1”, “3” and “7”.

Secure communication layer for a group

With a group key in place, the participating V2V devices may start efficient, secure communication. At the beginning of the group session, message counters are set up in each V2V module, such that a module’s own counter is incremented with every sent message and the counters of other modules are set to the

value obtained in the last successfully decrypted received message from that module. To send a message M from a module identified by a unique ID to another V2V module identified by ID', the message is encrypted using authenticated encryption, following the guidelines in Figure 133-14. The resulting ciphertext is encoded together with the message counter (if not already transmitted by the underlying protocol) and the result is used as a protocol payload. Upon reception, if decryption fails, the frame must be rejected.

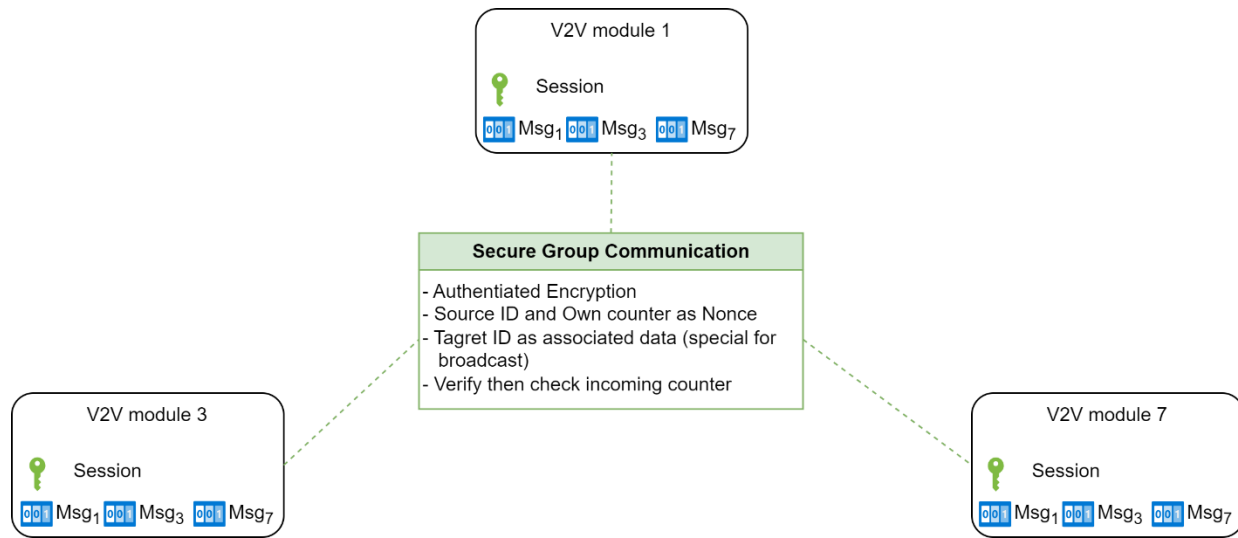


Figure 133-14 Securing communication in a group of V2V modules after key distribution in Figure 113-12.

3.2.2 Security considerations

This component has the following security objectives.

Objective	Attack vectors mitigated	Attack vectors non-mitigated	Remarks
V2V communication Confidentiality, authenticity, integrity & replay protection	Modification of the message being transmitted. Injection of new messages in the channel. Replay of previous messages.	Malicious interference from a (compromised) device that already holds the group communication key.	This residual risk is addressed by the Intrusion Detection Module, which ought detect abnormal behaviour of nodes.

	Sniffing of messages. Active man-in-the-middle attack		
Key establishment in a group of V2V modules	Long-term keys extracted from less than T V2V modules.	T and more V2V modules are compromised.	The threshold T is a configurable risk-performance trade-off parameter. To increase the difficulty of compromising T V2V modules without detection, the IDM module can be used to great effect (Sec. 3.5).

3.2.3 Integration considerations

Key provisioning. Each of the V2V modules must be provisioned with the threshold cryptography key share and credentials allowing for potin2point secure message exchange. In a real-world deployment, the provisioning of key shares must be done in a secure environment.

Communication complexity of the key establishment. The communication complexity due to the threshold-cryptography key establishment is significant. The V2V communication protocol should allow for a sufficient throughput while the group secret key is being established. The secure communication itself has a low overhead and is suitable even for constrained protocols.

3.3 Secure data collection with Bike Connection Box

Component code: **BCB**

Related to control IDs: **C01, C02, C03, C06, C07**

Table 5Based on the state of the art in IoT security, the Bike Connection Box (BCB) integrates several security mechanisms to address various security concerns, ensuring safe and reliable communication between autonomous cargo bikes and the TRACE platform (StreamHandler module). It implements controls related to data encryption, authentication, and access control to prevent unauthorized access. For instance, controls C02, C06 and C07 focuses on securing V2I communications over 4G/5G networks,

ensuring that critical vehicle and cargo data, such as location, speed, and delivery status, are protected from tampering or interception. The component described Section 3.2 complements this by ensuring the same security properties for V2V communication.

Bike-to-MASA communication will be based on MQTT. From a networking perspective, the bike will operate within a local network that limits access from external agents. MASA servers will securely forward data from the bikes to the TRACE platform to provide real-time data on delivery progress and vehicle operational status.

3.3.1 Specification

A description of the infrastructure, BCB, V2I and V2V can be found in D 4.1.

3.3.2 Security considerations

The security objective of BCB is to ensure **data integrity, confidentiality and authenticity** against data manipulation and spoofing attacks. This is ensured by using encrypted MQTT channels and robust authentication mechanisms. Key provisioning and access control to vehicles' embedded systems and MASA infrastructure are critical for ensuring secure communication. Attacks such as data spoofing, unauthorized access, or replay attacks are defeated by implementing TLS over MQTT and enforcing strong authentication protocols.

Assumptions for control effectiveness:

- Secure key provisioning mechanisms are in place.
- Strong access control policies for vehicle systems and cloud components.
- Real-time monitoring for any anomaly in data transmission.

3.3.3 Integration considerations

The integration of BCB security controls involves several components:

Devices: Embedded systems on vehicles, MASA infrastructure, and cloud-based TRACE platform (StreamHandler module).

Policies: Secure provisioning, encrypted data transmission, and access control protocols.

The main interaction between BCB and TRACE platform involves secure MQTT-based communication between via the MASA infrastructure. APIs need to facilitate real-time data sharing, allowing TRACE to

dynamically adjust vehicle routes and recharging decisions while ensuring security. The integration of embedded devices on the vehicles may be based on SPI, UART or similar technologies.

3.4 Blockchain SSID module

The Blockchain SSID module is responsible for implementing C28 and C29. Wallet generation must be secure, ensuring that the private key is not exposed to the open network. Once a wallet's private key is generated, it should be stored on the user's local machine. The user should have full control over the private key. Only the public key should be shared with the Trace ecosystem. Since the public key is widely available, it does not require strict security measures. However, it should remain intact during the process of storing it in the Trace database.

The dNFT module is responsible for integrating C44 during implementation. The data should be encrypted when being pushed to the blockchain. The encryption will be handled by the blockchain's core algorithm.

3.4.1 Specification

The blockchain technology leverages decentralized and secure modules to enhance identity management, asset tracking, and tokenization. It integrates (SSID and Dynamic Non-Fungible Token (dNFT) modules, focusing on secure data handling, privacy, and dynamic asset functionality.

SSID Module:

The SSID module facilitates decentralized identity management, empowering users to generate, store, and control their own digital identities without reliance on a centralized authority. It implements components such as C28 and C29, ensuring that private key generation is secure. The private key, which is essential for accessing and managing the wallet, remains local to the user's device, preventing exposure to networks or third parties. Only the public key is shared with relevant systems like the TRACE ecosystem, ensuring secure yet seamless interactions across the blockchain while maintaining user privacy and data control.

dNFT Module:

The dNFT module provides a framework for creating dynamic NFTs, which can evolve based on real-world events or data feeds. This module integrates C44, enabling NFTs to be updated or modified without compromising security or authenticity. The data that informs these changes is encrypted before being pushed to the blockchain and decentralized storage (IPFS), with the core blockchain algorithm managing encryption. The dNFT module supports real-time interactions and updates, making it ideal for applications

that require dynamic asset tracking, such as in supply chains, intellectual property, or digital collectibles. Together, these modules form a robust technology platform designed for secure, decentralized operations, with potential applications in areas such as supply chain management, digital identity, and tokenized assets.

3.4.2 Security considerations

The blockchain technology addresses several attack vectors by ensuring data security, identity privacy, and integrity through a combination of cryptographic controls, decentralized architecture, and strict key management practices. Below is an outline of how the technology defeats identified attacks and what needs to be guaranteed for controls to remain effective:

Private Key Exposure

Attack Defeated: Man-in-the-middle (MITM) attacks, key theft, and unauthorized access to private keys.

Mitigation Strategy: The SSID module stores private keys locally on the user's device, ensuring they are not exposed to open networks or third-party services. Only the public key is shared with the Trace ecosystem, limiting the attack surface for key theft.

Assumptions for Effectiveness: Users must maintain strict access control over their devices to prevent physical or remote unauthorized access. Proper key provisioning practices must be followed, including secure key generation and local storage mechanisms.

Data Integrity Attacks

Attack Defeated: Tampering or manipulation of data while stored or transmitted within the blockchain.

Mitigation Strategy: All data pushed to the blockchain through the dNFT module is encrypted using blockchain's core encryption algorithms, ensuring data confidentiality and integrity. Immutable records on the blockchain ensure that once data is written, it cannot be altered or tampered with.

Assumptions for Effectiveness: The cryptographic primitives (e.g., encryption algorithms, hash functions) used must be robust and secure against cryptanalysis. Access control mechanisms should ensure only authorized users can interact with or update dNFT data.

Replay and Transaction Forgery Attacks

Attack Defeated: Attackers attempting to resend valid transactions to gain unauthorized access or modify information.

Mitigation Strategy: The blockchain uses unique transaction identifiers (nonces or timestamps) to ensure that transactions cannot be replayed or forged. Public-key cryptography ensures that any changes or transactions initiated in the dNFT or SSID module can be verified by the signature, making forgery highly improbable.

Assumptions for Effectiveness: The blockchain's timestamping and nonce mechanism should remain accurate and tamper-proof. Access control to private keys and signing mechanisms must be properly enforced.

Sybil and Identity Theft Attacks

Attack Defeated: Creation of multiple fraudulent identities or unauthorized access to user identity.

Mitigation Strategy: The SSID module ensures that only verified users can interact with the Trace ecosystem by managing identities through public-private key pairs. Users have full control of their private keys, preventing unauthorized access or impersonation.

Assumptions for Effectiveness: Robust identity verification must be in place to prevent the creation of fake or duplicate accounts. Proper security measures need to be enforced on the user's device to prevent private key theft.

Out-of-Scope Attacks

Denial of Service (DoS) Attacks: These attacks, where a service can be rendered unavailable through flooding requests, are considered out of scope. The solution focuses on securing identity, transactions, and data rather than availability.

Side-Channel Attacks: Attacks like power analysis or timing attacks on local devices are also considered out of scope, as they target hardware vulnerabilities that this solution does not directly address.

Guarantees for Control Effectiveness:

Key Provisioning: Secure methods of generating and distributing keys must be employed, including using hardware security modules (HSMs) or trusted platforms.

Access Control: Strict access control policies must be enforced to ensure that only authorized users and systems can access private keys, sensitive data, or perform transactions.

Network Security: Secure communication channels (e.g., TLS) must be used to protect data during transmission and prevent eavesdropping or MITM attacks.

User Education: Users need to be educated on secure practices, including protecting their private keys and securing their devices against physical and software attacks.

By implementing these controls and adhering to the assumptions outlined, the technology effectively mitigates common attacks while maintaining user control over identities and data integrity across decentralized networks.

3.4.3 Integration considerations

Main components of the future implementation are the following:

Software Modules

SSID Module: Responsible for decentralized identity management, including wallet generation and private key management.

dNFT Module: Handles the creation, storage, and management of dynamic NFTs, ensuring secure data updates and transactions.

Devices

User Devices: Smartphones, tablets, or computers that users will use to generate and manage their identities and dNFTs.

Blockchain Nodes: Servers or cloud infrastructure that maintain the blockchain network, validate transactions, and store the blockchain ledger.

Policies

Access Control Policy: Defines permissions for users to interact with the SSID and dNFT modules, ensuring that only authorized users can access or modify data.

Data Encryption Policy: Specifies the encryption methods to be used for securing data transmitted to and stored on the blockchain.

Integration in the TRACE Architecture

Mapping: The SSID and dNFT modules will connect to the core TRACE architecture, facilitating identity verification and dynamic asset management. User devices will communicate with the SSID and dNFT modules through secure APIs, allowing users to perform operations like wallet management and NFT transactions. Blockchain nodes will serve as the backbone of the TRACE architecture, handling all transactions and ensuring data integrity across the network.

APIs: The TRACE ecosystem will expose RESTful APIs for user devices to interact with the SSID and dNFT modules. These APIs will handle requests for identity verification, wallet management, and NFT creation/updating. The dNFT module will provide endpoints for pushing and retrieving encrypted data, ensuring data integrity and confidentiality.

Communication: All communication between user devices and the SSID/dNFT modules will be secured using HTTPS and encryption protocols to prevent eavesdropping and data tampering. Interaction with blockchain nodes will involve a consensus mechanism to validate transactions, ensuring all changes are agreed upon by the network before being recorded.

Summary

In summary, the integration of the SSID and dNFT modules into the TRACE architecture involves defining software modules, user devices, and policies, along with establishing secure APIs and communication channels. This setup enables efficient and secure identity management and dynamic asset tracking while ensuring the integrity and privacy of user data within the TRACE ecosystem.

3.5 Intrusion Detection Module and Events Management

Component code name: IDM

Related to control IDs: C83

The **Intrusion Detection Module (IDM)** is an important part of TRACE as it is designed to protect the TRACE system from various types of cyber threats by monitoring the communications and data streams coming from the vehicles' sensors and not only. As TRACE will use vehicles for the deliveries of shipments, it needs a shield of protection against attacks that can damage and compromise the integrity of the data and even the security of critical infrastructures in our case the transport and logistics chain.

The IDM will be an automated observatory module, always receiving sensor/control data to detect anomalies. When any potential threat is detected, IDM not only flags the event but causes a cascade of actions which might reduce or eliminate it. Such reaction and prevention functions of the module will guarantee security and stability for the system.

This should make the IDM application observe, on the backend level, the messages sent by the vehicles during operation and, thus, allow monitoring in real time. It would also share data with the platform through message queues, thus making the flow of data possible across different system components.

The main role of IDM would be that it checks the network to scan the incoming data and then ascertains whether the behaviour it witnesses is normal or it depicts the possibility of intrusion. It can either be designed as a static system using some rules and thresholds for anomaly detection or a dynamic system which builds its knowledge over time. It allows the module to become agile and precise through dynamic learning from the data processed for better detection in real-world situations.

3.5.1 Specification

Key Functions

1. **Real-time data monitoring:** Real-time data monitoring: IDM is supposed to observe and analyse data from messages or raw sensor data generated by the vehicle. It then uses patterns in that sensor data to make inferences about possible anomalies, maybe something suspicious or some attempted attack.
2. **Anomaly Detection:** This provides data classification to filter normal traffic from potentially malicious activities. Through machine learning algorithms, it becomes dynamic and develops the ability to change with time due to new streams of data, hence enhancing its response to new patterns of intrusion.
3. **Communication to/from the platform-data** can be exchanged without latency between the platform and IDM. This will enable the system to be effective both at the local level, at vehicle and sensor locations, and in the cloud while maintaining stable performance.
4. **Intrusion Detection and Response:** The instant the module detects something anomalous, it sends a response to mitigate the probable effects in case of intrusion. This may include isolation of the sensors or systems affected and instigating broader actions.

Technology Overview

The technology behind IDM supports different techniques to process big streams of data, detect anomalies, and handle events. These techniques are analysed as under:

1. **Data stream processing:** The IDM technology processes continuous streams coming from sources so that it can analyse and assess events. This provides immediate identification of anomalies and shortens the gap between threat detection and mitigation.
2. **Classification methods:** The machine Learning techniques would help the system to differentiate between normal and abnormal behaviour by making predictions and triggering alerts.
3. **Data from multi-source integration:** In our approach, we can correlate data from multiple sources so that their cross-validation ensures higher efficiency with regard to the accuracy of the predictions.

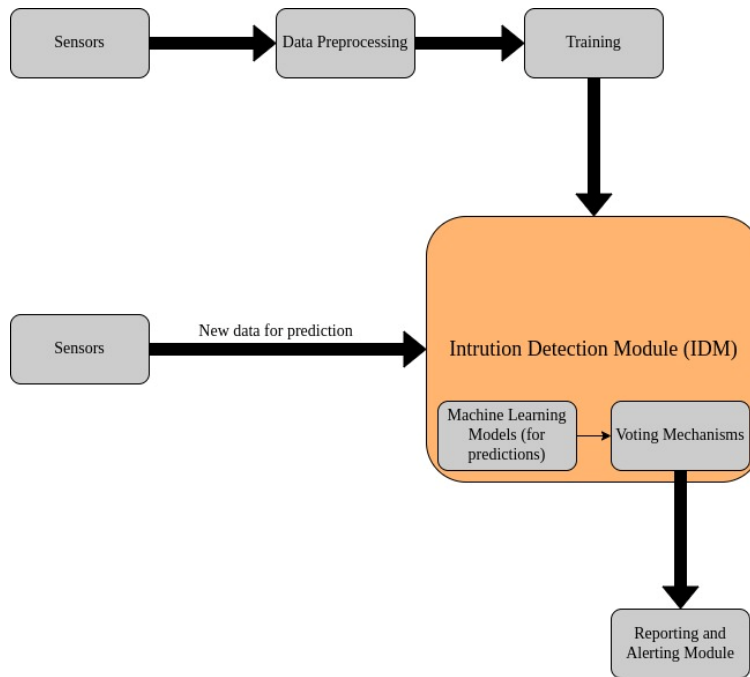
Architecture and Dataflow

The IDM architecture is designed to withstand a continuous flow of data without compromising the speed of the response dealing with security incidents. The group operates in an interdependent system; therefore, its data flow will have the following:

Data Collection and Preprocessing: Data is collected from a series of devices and sensors deployed in the system. After that, it is pre-processed to ensure the correct quality and relevance of data for intrusion detection purposes. Preprocessing entails cleaning, normalizing, and filtering of data that help the detection engine with emphasizing the most relevant information.

The Anomaly Detection Engine uses advanced methods to many different classification techniques to detect deviations from normal behaviour. It either uses predefined detection rules or may be set up to adapt its models over time to handle new forms of threats. By continuously analysing streams of data, it ascertains that the prejudicial data is detected well in advance.

Incident management-when the IDM detects intrusion, it will, based on predefined configuration, automatically initiate those actions aimed at minimizing the impact of the intrusion. Such actions may range from device isolation and blocking malicious communication channels to system-wide recovery actions. The IDM interacts with other modules within the platform to make such a response comprehensive and effective in minimizing disruption potential of the detected intrusion.



- **Sensors:** The whole system starts working on the data input it receives from vehicle sensors, communication infrastructure, or any other member element in the network. It is this sensor information base that dictates identification with respect to potential threats or suspicious activities.

- **Preprocessing:** Before the use of data, it includes cleaning, hence removing noise or irrelevant parts, and preparing data for further processing. This step is very important because the data should be in a format that fits handling models efficiently.
- **Training:** This is the second level, where models will actually be trained. Training here could involve the use of historical data that might include normal and abnormal behaviour in order to teach the system how to distinguish between safety and potentially harmfully related activities. It is expected that machine learning algorithms would undergo a training process so that they learn patterns of the data. In order to train Machine Learning (ML) models to detect anomalies in logistic scenarios, historical data containing normal or non-normal data will be used. We will compile historical datasets, such as sensor data, vehicle telemetry and route information. These data will be cleaned, standardized and structured for effective training. If there are labels in the dataset (like normal or anomalous) we can use supervised training algorithms that can learn the pattern from historical data. In the case where there are no labels in the dataset unsupervised learning models will be used to identify outliers or abnormal events.

A hybrid model can also be used in training, first with an unsupervised model that flags the intrusions and then with a supervised model that does the training. The trained intrusion detection models can be integrated into both the TRACE cloud platform and the sink manager for continuous monitoring of live data. Using the metrics such as f1 score, precision and recall we can check the efficiency of our model and make it more efficient. The trained model will validate using recent data to check its efficiency and precision by measuring its ability to find intrusion without generating false alarms. There will be the possibility to train periodically after collecting new data to retrain when needed.

- The core of the whole diagram is the IDM itself. This module will be responsible for real-time monitoring and prediction. The machine learning model hosted in the IDM will analyse the new incoming data to find anomalies or probable intrusion activity after being trained.
- **Machine Learning Models:** Trained models used for making the prediction on each activity coming from sensor(s) preprocessing.
- **Voting Mechanisms:** IDM implements an ensemble method where decisions are taken en masse by several models. Furthermore, it makes use of a voting mechanism so that decisions are not taken solely by one model but rather by the plurality of models. This approach increases the accuracy of the whole detection process.

- **New Data to Predict:** Newly gathered sensor data will be fed directly into the IDM to make the prediction in real time. Thus, it will allow for the continuous monitoring of the system's security status.

3.5.2 Security considerations

The **Intrusion Detection Module (IDM)** is designed to proactively identify and mitigate various kinds of cyber threats regarding data integrity, confidentiality, and availability of the platform. This technology makes use of advanced machine learning algorithms as part of an effective way to handle all common attack vectors. Below, we outline how these technologies help defeat identified attacks and the assumptions necessary for effective control.

3.5.3 Defeating Identified Attacks

1. **Malware Attacks:** The classifiers are trained with a labelled dataset to identify patterns indicative of malware behaviour in historical attack data. These models would utilize the derived features from incoming traffic and system logs to determine benign activities versus malicious activities in nature, in order to prevent malware from penetrating into the system.
2. **Denial of Service Attacks:** An IDM can detect, while monitoring traffic patterns and resource utilization, abnormal spikes in network activity that usually raise suspicion of a DoS attack. The ensemble voting mechanism improves accuracy to a sufficient level for automatic triggering of the mitigation action-like rate limiting or blocking offending IPs-by the system itself.
3. **Anomalous Behaviour Detection:** This is designed to identify deviation from normal behaviour; such would then point towards a potential intrusion based on previously calculated baselines. This is very effective, especially in zero-day attacks, where there is no prior knowledge of the attack.
4. **Data Breaches:** Multi-classifier integration offers hard-on-the-nail protection against data breaches, since suspicious activities in data access and transfer are tattled on through real-time flagging. The voting mechanism amalgamates all the model predictions to reduce false negatives and generally enhances the security posture of the TRACE platform.

They can also be trained on phishing attack datasets to enable the system to identify patterns related to phishing messages or suspicious user behaviour and alert the administrators of a possible phishing incident.

3.5.4 Effective Control

To make IDM effective to handle all these identified attacks, the following assumptions have to be in place:

1. **Key Provisioning:** Key provisioning processes should be defined in a secure manner, since protection of sensitive data is a must. Generation, distribution, and storage of the keys need to be done in such a way that unauthorized access is avoided, which might be supported by hardware security modules or dedicated key management services.
2. **Access Control:** The IDM should have different controls enforced on its models, data sources, and communication interfaces. This would mean that role-based access control is granted in locking access to sensitive data and functions against unauthorized persons.
3. **Data Integrity:** Checks must be performed to guarantee that utilized datasets are genuine, reliable, and resistant against adversarial tampering of training data with the aim of decreasing model accuracy.
4. **Secure Communication:** Communication channels in IDM should be encrypted. It should contain secure APIs for model predictions and event reporting; also, sensitive information could not be intercepted during the transmission.
5. **Effective Attack Scope:** While IDM is capable of handling most kinds of variant attacks, certain targeted and sophisticated ones may fall out of scope. This can also include APTs or insider threats that depend upon proper privileges being available for access. The system will constantly require monitoring in order to keep pace with evolving threats through periodic updates in the training datasets.
6. **Regular Model Updates:** Models need periodic updates by re-training with the state-of-art data for efficient defences against novelty attack vectors. This includes the new examples of malicious activity included in the training dataset and the fine-tuning of algorithms to adapt against shifting attack patterns.

3.5.5 Integration considerations

Main Components of the Future integration are the following:

- Software Modules:

1. **Intrusion Detection Module (IDM)** : This module accommodates the major machine learning algorithms for anomaly detection and classification of network traffic. It also comprises a variety of data preprocessing techniques and feature extraction that increase the accuracy of the models. It consists of an ensemble of model predictions using a voting mechanism that provides predictable improvement in the reliability of the detection.
 2. **Data Preprocessing Module**: It cleanses and transforms incoming data. This module will resolve the issues of missing values, normalize data, and balance classes by using different techniques.
 3. **Training and Evaluation Module**: It governs the training process for the machine learning models. This module will have a scheduled retraining of models with the latest dataset to adapt to new kinds of threats.
 4. **Reporting and Alerting Module**: This module will issue an alert and report the result of the detection and will also be integrated into the TRACE dashboard for real-time monitoring.
- **Devices**:
 1. **Sensors**: Various sensors will be installed in the TRACE network to gather data from network traffic, user behaviours, system logs and the sensors will relay the collected data to IDM for its analysis.
 2. **Central Processing Unit (CPU)**: The IDM and its components shall be hosted on a dedicated server or cloud instance. This shall host the training of models, data storage, and processing.

3.5.6 Integration into TRACE Architecture

The following illustrates how the IDM and its components integrate into the TRACE architecture:

Component	TRACE Interaction	Description
Intrusion Detection Module (IDM)	Central Component	The IDM acts as the central security component, analysing data from various sensors and sending alerts to the TRACE dashboard.
Data Preprocessing Module	Data Flow Management	Collects raw data from sensors, processes it, and feeds it into the IDM for analysis.

Training and Evaluation Module	Feedback Loop	Uses historical data and detection results to retrain models, improving their accuracy over time.
Reporting and Alerting Module	User Interface Integration	Sends alerts and reports to the TRACE dashboard for real-time monitoring by operators.
Sensors	Data Collection Layer	Deploys across the TRACE network to gather data for intrusion detection.
Central Processing Unit (CPU)	Core Computing Resource	Hosts the IDM, processing data and running machine learning algorithms.

3.6 End-to-end encryption of data in transit and at rest

Component code: DataStoreCrypt

Related with Control IDs: C07 and C44

The DataStoreCrypt component is a critical enhancement to the security architecture of the TRACE platform, which handles vast amounts of data for logistics companies. This component is designed to provide robust end-to-end encryption, ensuring data remains protected both while at rest in storage and during transmission across the platform's distributed systems. DataStoreCrypt provides a comprehensive security solution that aligns with modern data protection standards by encrypting data in transit, including messages broadcasted in Kafka, and securing data at rest, such as files stored in MongoDB.

In the logistics industry, data such as shipment details, customer information, and operational analytics are highly sensitive and valuable. Any breach or unauthorized access could have severe financial, reputational, and operational consequences for companies. Moreover, logistics is one of the markets where certain stakeholders are highly concerned about the potential for uncontrolled data leakage to competitors, making direct user-controlled end-to-end encryption crucial in this context. DataStoreCrypt addresses these risks by enforcing encryption at every stage of data handling. Even if data were to be accessed by unauthorized parties, the encryption would render it unreadable, safeguarding the integrity and privacy of the logistics data.

Moreover, TRACE's distributed architecture, which relies on real-time data exchanges between various stakeholders, requires an efficient yet highly secure means of transmitting data. By integrating Kafka with strong encryption protocols and effective key management, DataStoreCrypt ensures that data remains protected as it moves through the system, preventing interception or tampering during its journey from source to destination.

DataStoreCrypt has been developed with one specific goal in mind, to enhance the trust between TRACE and its clients. In a sector where data security is paramount, this component offers a powerful, scalable solution for safeguarding critical logistics data, allowing TRACE to deliver secure, reliable services without compromising performance.

3.6.1 Specifications

- DataStoreCrypt with MongoDB

The architecture for the DataStoreCrypt component as depicted in Figure 1 is centred around securely managing the encryption and decryption of data as it is stored in and retrieved from MongoDB. This architecture leverages a combination of a Key Encryption Service (KES), HashiCorp Vault², MiniIO³, and MongoDB, ensuring that sensitive data is always encrypted at rest.

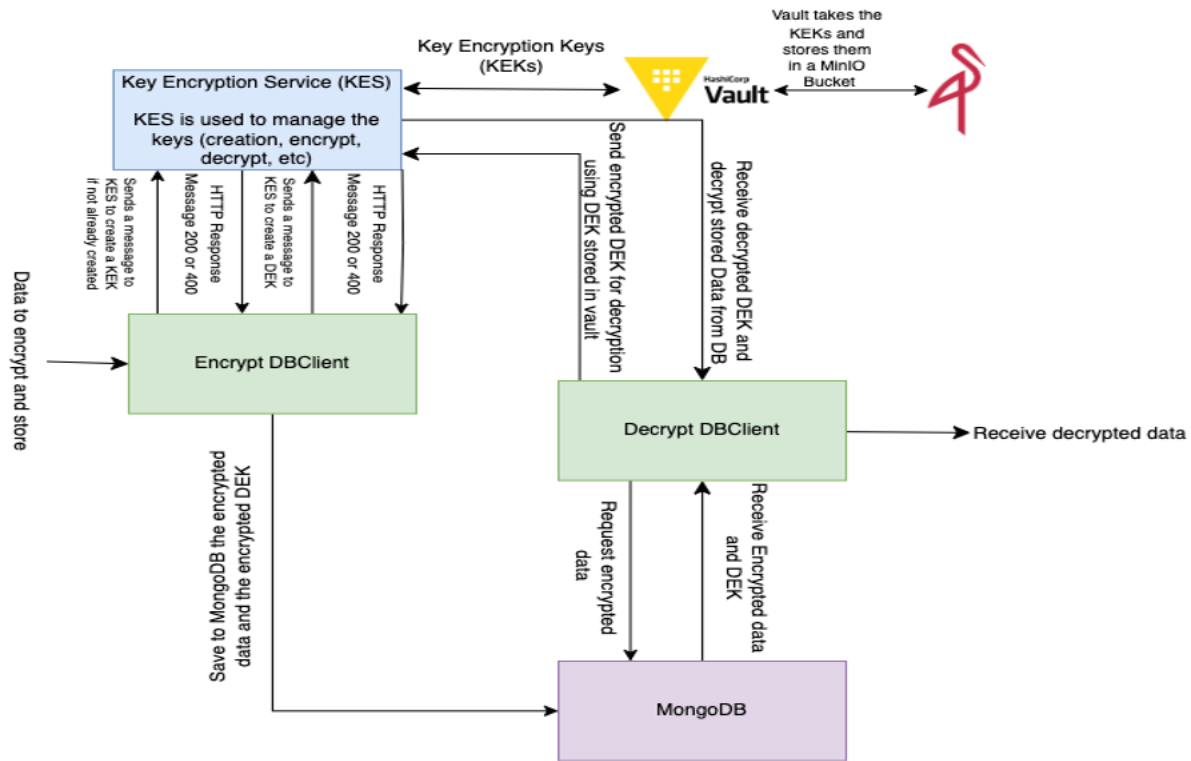


Figure 6: DataStoreCrypt architecture and workflow for encrypting/decrypting data at rest using MongoDB

Key Components and Flow:

1. Key Encryption Service (KES): The KES manages all cryptographic keys involved in the process. It is responsible for creating, encrypting, decrypting, and managing both the Key Encryption Keys (KEKs) and Data Encryption Keys (DEKs). The KES communicates with the other services and initiates the encryption process by generating a DEK for each piece of data that needs to be encrypted.

² <https://www.vaultproject.io/>

³ <https://min.io/>

2. **Encrypt DBClient:** The Encrypt DBClient is the component responsible for encrypting data before it is stored in MongoDB. It first checks whether a KEK (Key Encryption Key) is already created by sending a message to the KES. If no KEK exists, the KES will generate one. The Encrypt DBClient then sends a request to the KES to create a DEK (Data Encryption Key) that will be used to encrypt the data itself.

Once the DEK is generated and received, the Encrypt DBClient encrypts the data using this key and stores both the encrypted data and the encrypted DEK into MongoDB. The encrypted DEK is stored alongside the encrypted data to allow for future decryption.

3. **MongoDB:** MongoDB serves as the secure data storage in this architecture. It holds the encrypted data and the corresponding encrypted DEK for each entry. This ensures that even if the database is compromised, the data remains secure and inaccessible without the keys stored elsewhere.

4. **Decrypt DBClient:** The Decrypt DBClient is responsible for retrieving and decrypting the data. When a request is made to retrieve data, the Decrypt DBClient fetches the encrypted data and its associated encrypted DEK from MongoDB. It then sends the encrypted DEK to the KES for decryption.

The KES decrypts the DEK and returns it to the Decrypt DBClient, which then uses the decrypted DEK to decrypt the stored data. Once the data is decrypted, it is returned to the requesting service in its original form.

5. **HashiCorp Vault and MinIO:** The HashiCorp Vault stores the KEKs (Key Encryption Keys) securely. These KEKs are used to encrypt and decrypt the DEKs. Vault acts as a secure storage for the cryptographic material that ensures the DEKs are only accessible when needed. The KEKs are stored in a MinIO bucket managed by the Vault, providing an additional layer of security and storage management.

Key Workflow:

- When new data is submitted for storage, the Encrypt DBClient requests a DEK from the KES and encrypts the data. The encrypted data and the encrypted DEK are then stored in MongoDB.
- For decryption, the Decrypt DBClient fetches the encrypted data and encrypted DEK from MongoDB, sends the DEK to the KES for decryption, and then decrypts the data using the returned DEK.

This architecture ensures that sensitive data is protected at all stages—both during storage (at rest) in MongoDB and during transit. The separation of concerns between encryption, key management, and data storage adds multiple layers of security, making it a robust solution for securing sensitive data.

- DataStoreCrypt with Kafka

Additionally, the DataStoreCrypt component that is depicted in Figure 2, also secures data while it is in transit in the Kafka message bus. This setup ensures that any sensitive data being transmitted between services remains encrypted and secure, preventing unauthorized access during data transfer. Crucially, the authorization configuration for the use of decryption keys remains in the hands of the individual users. The integration of a Key Encryption Service (KES) with Kafka Producers and Consumers enables secure encryption and decryption of messages while in transit.

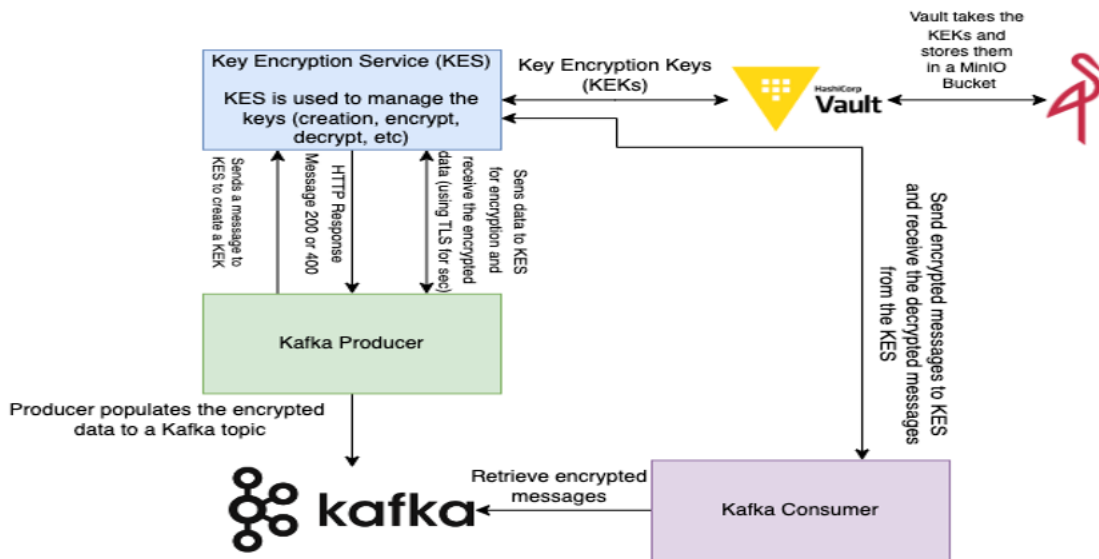


Figure 2: DataStoreCrypt architecture and workflow for data in-transit using Apache Kafka

Key Components and Flow:

1. Key Encryption Service (KES): As in the previous architecture for data-at-rest, the KES is responsible for managing the cryptographic keys, including the Key Encryption Keys (KEKs) and Data Encryption Keys (DEKs). It handles the creation, encryption, and decryption of keys, ensuring that messages are securely encrypted before being sent through Kafka and properly decrypted upon retrieval.
2. Kafka Producer: The Kafka Producer is responsible for generating messages and encrypting them before publishing to Kafka topics. When a message is prepared for publishing, the producer sends a request to the KES to generate or retrieve the appropriate encryption keys (DEKs) for securing the data. The producer then encrypts the message using the DEK and publishes the encrypted data to a Kafka topic.

The producer ensures that all messages are transmitted in an encrypted form, mitigating the risk of interception or tampering while the data is in transit between services or across distributed systems.

3. Kafka: Apache Kafka serves as the message broker in this architecture, where encrypted messages are stored temporarily in topics. Kafka enables the seamless transmission of messages between producers and consumers in a highly distributed environment, ensuring high throughput and low latency. However, to ensure security, the data within Kafka is always encrypted by the producer.

4. Kafka Consumer: The Kafka Consumer retrieves the encrypted messages from Kafka topics for further processing. Upon receiving the encrypted data, the consumer forwards the message to the KES for decryption. In order to identify the correct KEK for decryption the key name is sent as a key parameter in the Kafka message which is then used by the consumer to decrypt the message and return it in its original form for downstream applications or services.

5. HashiCorp Vault and MinIO: Just as with the data-at-rest architecture, the HashiCorp Vault plays a critical role in securely storing the KEKs (Key Encryption Keys). Vault ensures that these keys are safely stored and only accessible to authorized services through the KES. The KEKs themselves are stored in a MinIO bucket, which acts as a secure storage medium for key material. This guarantees that encryption keys are properly secured during their lifecycle.

Key Workflow:

- The Kafka Producer encrypts the data using DEKs generated by the KES and sends the encrypted message to a Kafka topic.
- Kafka acts as the broker, handling the encrypted messages in a secure, distributed manner.
- When a Kafka Consumer retrieves the encrypted message, it sends the DEK to the KES for decryption.
- The consumer then decrypts the message using the returned DEK and processes the original message as needed.

This architecture ensures that all data transferred between services via Kafka is encrypted, safeguarding it from unauthorized access or interception during transit. The integration of Kafka with KES and Vault provides a powerful, scalable, and secure messaging system within the TRACE platform, critical for handling the sensitive data of logistics companies.

3.6.2 Security Considerations

1. Key Management:

Key Storage Security: Ensuring that the Key Encryption Keys (KEKs) managed by HashiCorp Vault and stored in a MinIO bucket are encrypted and access controlled. HashiCorp Vault protects keys through a robust mechanism of sealing and unsealing. When the Vault starts, it is sealed and requires a threshold of unseal keys to unlock it. These keys are distributed among trusted personnel, ensuring that unauthorized users cannot access sensitive data.

Vault encrypts all data at rest, meaning that even if the storage is compromised, the actual data remains protected. Access control is enforced through fine-grained policies that limit who can access KEKs, utilizing various authentication methods such as tokens and Identity and Access Management (IAM) roles.

While Vault mitigates risks like unauthorized access and data breaches, it's important to be aware of potential threats such as social engineering attacks aimed at obtaining unseal keys. To minimize risk, unseal keys should be securely managed, potentially using hardware security modules (HSMs) or distributed among trusted individuals.

By implementing these security measures, we can ensure the integrity and confidentiality of the encryption keys and the overall encryption scheme.

Key Rotation: Rotating mechanism for both KEKs and DEKs to reduce the impact of potential key compromise is mandatory. Automatic key rotation policies ensures that the old keys can decrypt previously encrypted data.

Key Expiry and Revocation: KEKs and DEKs should have an expiration policy to limit the exposure window. Mechanisms for key revocation in case of suspected compromise will also further limit the exposure.

2. Secure Transmission:

TLS/SSL for Kafka: Communication between Kafka Producers, Consumers, and the KES should be conducted over a secure channel using TLS (Transport Layer Security). This will prevent man-in-the-middle (MitM) attacks where attackers could intercept or tamper with messages during transit.

HTTP Requests to KES: All HTTP requests made to the KES for key generation, encryption, or decryption should be encrypted using TLS. This will protect the data from being intercepted in transit between components.

3. Access Control:

Role-Based Access Control (RBAC): Only authorized services should have access to KES for creating, encrypting, or decrypting keys. Ensuring that roles and permissions are strictly defined will limit access based on the principle of least privilege.

4. Audit and Monitoring:

Key Usage Logging: Keeping an audit trail of all key-related activities (e.g., key creation, encryption, decryption) and monitoring for suspicious or anomalous behaviour will further enhance security. Real-time alerting mechanisms will detect potential security breaches.

Data Access Monitoring: Monitor access patterns to encrypted data in MongoDB and Kafka topics. Alerts should be triggered on any unusual access activity, such as a spike in data requests that could indicate an attack.

5. Backup and Recovery:

Encrypted Backups: Ensure that any backups of encrypted data and keys are also encrypted. This guarantees that if backup data is accessed, it cannot be exploited without the proper keys.

Key Recovery: In the event of key loss, there should be a secure mechanism in place to recover keys from backups. This could involve multi-factor recovery methods to ensure keys are not improperly restored.

3.6.3 Integration Considerations

Multiple aspects need to be considered in the integration of DataStoreCrypt in the TRACE platform, as listed below:

Compatibility with Existing Systems:

Interface Design: It is necessary to ensure the DataStoreCrypt component interfaces seamlessly with existing systems like MongoDB and Kafka. Define clear and consistent APIs for encryption and decryption to maintain compatibility and ease of integration.

Protocol Compatibility: Verify that the component adheres to the communication protocols used in the TRACE platform (e.g., REST, gRPC) and integrates with any existing message queuing or processing frameworks.

Configuration Management:

Centralized Configuration: It is necessary to store configuration settings (such as encryption keys, endpoint URLs, and security parameters) in a centralized configuration management system. This simplifies updates and changes, ensuring that all services can access the necessary parameters.

Environment-Specific Settings: Having configurations for different environments (development, testing, production) will ensure that sensitive keys and endpoints are not hard-coded and are appropriately managed.

Security Policies and Access Control

Access Control Policies: Enforcing access control policies for the DataStoreCrypt component will ensure that only authorized services and users can access encryption keys and perform encryption or decryption operations.

Testing and Validation:

Integration Testing: Integration testing will validate that DataStoreCrypt works as expected within the TRACE platform. This includes testing all interfaces and ensuring compatibility with other components.

Performance Testing: Testing the component under load in order to ensure that it meets performance requirements, especially in terms of response time for encryption and decryption operations.

Monitoring and Logging:

Performance Metrics: Implementing monitoring solutions will help to track the performance of the DataStoreCrypt component, including latency, throughput, and error rates. This can help in identifying bottlenecks or issues.

Audit Logs: Maintaining detailed logs of all encryption and decryption operations, including timestamp, user/service requesting the operation, and the status of the operation. This is essential for compliance and auditing purposes.

Future Scalability:

Load Balancing: If high traffic is expected, implementing load balancing would be crucial for KES to ensure that it can handle multiple requests efficiently without becoming a bottleneck.

By considering these integration factors, we ensure that the DataStoreCrypt component not only fits well within the TRACE platform architecture but also enhances its security and performance in handling sensitive logistics data.

4 Virtual Cockpit

This Section is dedicated to the advanced UX aspects of TRACE, addressed in Task 4.7. In particular, the Virtual Cockpit is one of the user interfaces developed in the context of the TRACE project to introduce the VR technology and its benefits to the smart transportation sector. The Virtual Cockpit is an immersive application presented as a standalone module that can communicate with the TRACE platform through the StreamHandler. It comprises a virtual dashboard where the user can monitor transportation vehicles remotely through video/audio streaming and GPS signal tracking. Additionally, the user can intervene to the vehicle's roundtrip in case of detected emergency or unwanted behaviour through sending one of the predefined emergency signals from the virtual dashboard to the corresponding vehicle. The nature of the communication between this standalone and external (with respect to the TRACE platform) module with the vehicles makes it necessary for the consortium to analyse and plan ahead in terms of the security of the streaming information and the privacy of the communication.

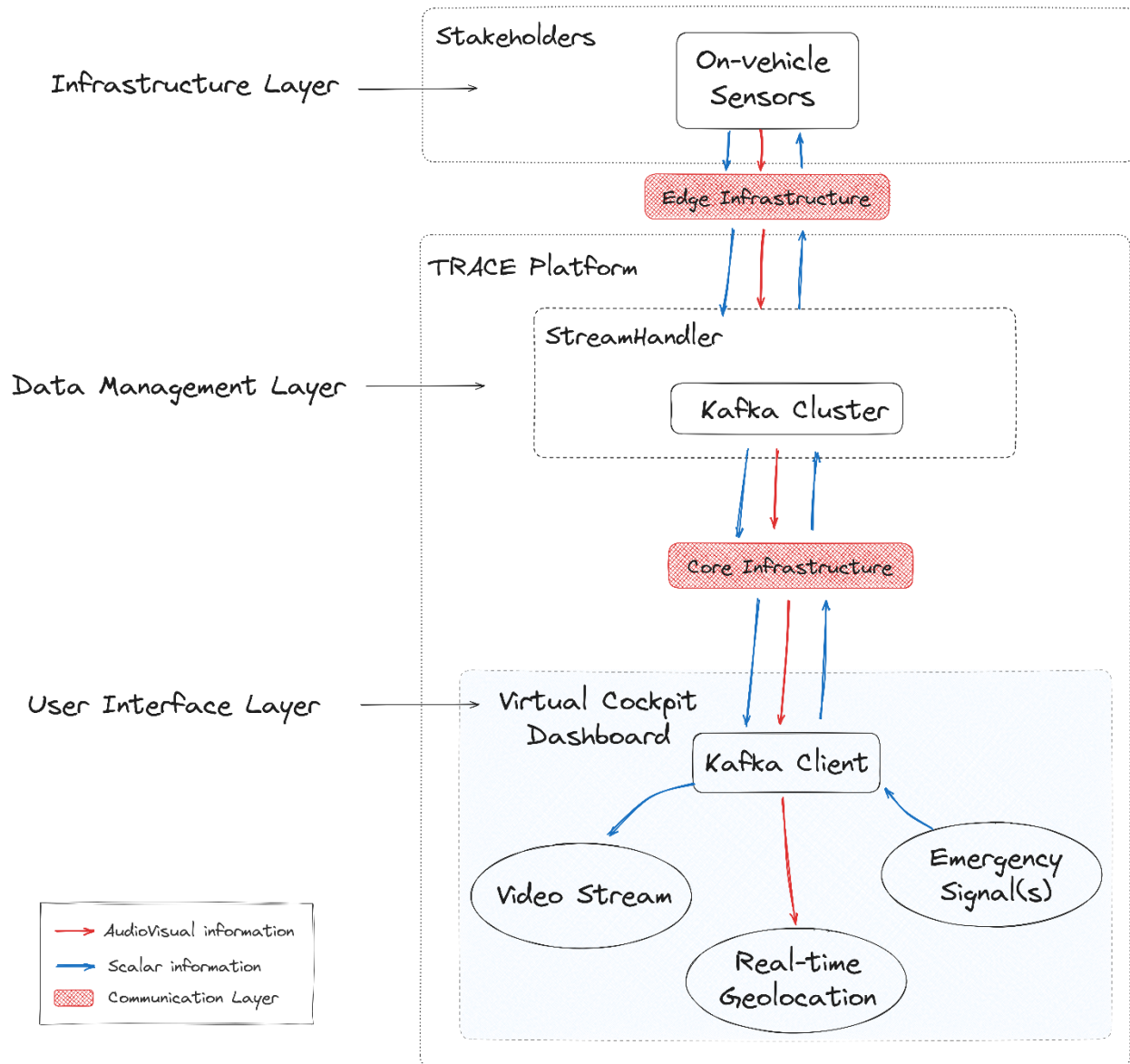


Figure 4-1: Overview of the Virtual Cockpit Module in TRACE platform

4.1 Description of the component

4.1.1 Key Features

Video Stream Reception and Display in VR

1. **High-Definition Video Streaming:** The application receives real-time video feeds from the supported vehicle's onboard camera(s), supporting up to high-definition (HD) quality to provide clear and detailed visuals in a VR environment.
2. **Low Latency:** Ensures minimal delay between the live feed and its display in VR to enable timely decision-making and situational awareness.

GPS Signal Reception and Real-Time Mapping in VR

3. **Real-Time GPS Tracking:** Continuously receives GPS data from the supported vehicle to track its location accurately.
4. **Virtual Map Integration:** Projects the GPS coordinates onto a detailed 3D virtual map within the VR environment, displaying the vehicle's current position, speed, and heading.
5. **Interactive Route Monitoring:** Allows operators to monitor the vehicle's adherence to the planned path in an immersive 3D space.

Emergency Functionality

6. **Predefined Emergency Codes:** The VR cockpit can send predefined commands to the unmanned vehicle for critical actions, such as:
 - a. **Return to Base:** Command to instruct the vehicle to navigate back to its home base or starting point.
 - b. **Emergency Stop:** Immediate halt of all vehicle operations.
 - c. **Land:** For aerial vehicles, commands to land safely at the nearest suitable location.

User Interface in VR

Dashboard Layout

1. **Virtual Control Panels:** Interactive panels within the VR environment that display live video streams, maps, and control options. Operators can manipulate these panels.
2. **3D Map Display:** An interactive 3D map showing the real-time GPS position of the vehicle with additional layers for route planning and geofencing, integrated into the VR scene.
3. **Status Indicators:** Virtual HUD (Heads-Up Display) elements showing vital statistics like battery level, signal strength, speed, altitude (for aerial vehicles), and other relevant metrics.

Notifications and Alerts

4. **Immersive Alerts:** Visual and audio alerts integrated into the VR environment for immediate notification of critical events such as loss of signal, low battery, or deviation from the planned route.
5. **Log and History:** A virtual console displaying a log of all commands sent, received data, and events for post-mission analysis and review.

System Requirements

Hardware

1. **High-Performance Servers:** For processing video streams and handling real-time data.
2. **VR Devices:** Compatible with major VR headsets (e.g., Oculus Rift, Oculus Quest) to render the virtual cockpit.
3. **Reliable Network Connection:** Ensures seamless data transmission between the unmanned vehicle and the virtual cockpit.
4. **Powerful GPUs:** Necessary to render high-quality VR scenes smoothly and responsively.

Software

5. **Operating System:** Compatible with Windows.

6. **VR Software Development Kit (SDK):** Integration with VR SDKs (e.g., Oculus SDK) for developing and running VR applications.
7. **Video Streaming Software:** Supports low-latency video processing and display in VR.

The Virtual Cockpit offers an immersive and intuitive interface for managing unmanned vehicles, enhancing operational efficiency, situational awareness, and safety through a highly interactive and engaging experience.

4.2 Virtual Cockpit Data Streams

Video stream: The Virtual Cockpit module is designed to efficiently handle the reception, processing, and streaming of video data from the supported vehicles to a Virtual Cockpit application. The architecture leverages Apache Kafka for robust and scalable data streaming. The system starts with a supported vehicle, equipped with high-definition cameras that capture live video. This video encoder then compresses this video using codecs like H.264 to optimize bandwidth usage. The encoded video is transmitted over a secure network to the TRACE platform's StreamHandler. Upon reaching the platform, the network receiver module accepts the incoming video stream. The StreamHandler's Kafka Producer then publishes the video data to a Kafka topic dedicated to the specific vehicle or mission. The Kafka Broker manages these topics, ensuring that the video data is available for consumption by multiple clients if needed. The Virtual Cockpit application includes a Kafka Client that subscribes to the relevant Kafka topic to receive the video stream. The client consumes the video data, ensuring low latency and high reliability. This video data is then decompressed by a video decoder. Finally, the VR renderer integrates the video stream into the Virtual Cockpit's 3D environment, providing an immersive experience for the operator.

Geolocation stream: The GPS stream is a crucial component designed to handle the reception, processing, and streaming of GPS data from unmanned vehicles to a Virtual Cockpit application. This module also utilizes Apache Kafka for efficient and scalable data streaming. The supported vehicles are equipped with GPS receivers to capture real-time location data. This GPS data is encoded and transmitted over a secure network to the TRACE platform's StreamHandler, where the network receiver accepts the incoming GPS data stream, while the respective Kafka Producer publishes the GPS data to a dedicated Kafka topic associated with the specific vehicle. The Kafka Broker manages these topics, ensuring that the GPS data is readily available for consumption by multiple clients as needed. The Virtual Cockpit's Kafka Client subscribes to the relevant Kafka topic to receive the GPS stream, which consumes the GPS data ensuring low latency and high reliability. This GPS data are subsequently processed and mapped onto a detailed 3D virtual map within the VR environment. The map displays the vehicle's current position, speed, and heading, providing operators with real-time situational awareness.

Emergency response stream: The emergency response stream is a vital component designed to handle the reception, processing, and streaming of emergency commands to the supported vehicles. The operators can initiate predefined emergency commands, such as "Return to Base," "Emergency Stop," or "Land." These commands are entered through the VR interface using hand gestures or VR controllers for intuitive interaction. Upon command initiation, the Virtual Cockpit's Kafka Producer encodes and publishes the emergency command to a dedicated Kafka topic corresponding to the specific vehicle or mission. The Kafka Broker manages these topics, ensuring that the emergency commands are reliably and

efficiently distributed to the appropriate recipients. On the TRACE platform, the StreamHandler's Kafka Client subscribes to the relevant Kafka topic and receives the emergency command. The client consumes the command and passes it to the network transmitter module, which securely sends the command to the unmanned vehicle over the network. The supported vehicles are equipped with command receivers that continuously listen for incoming commands. Upon receiving an emergency command, the vehicle's onboard system decodes the command and executes the predefined action immediately, ensuring swift and appropriate responses to critical situations.

4.3 Security & Privacy Options

TRACE's Virtual Cockpit has several key security and data privacy concerns that must be addressed to ensure the system is robust, secure, and respects user privacy. Starting from the data transmission security, 3 challenges have been identified and will be addressed: a) encryption – all data transmissions (video streams, GPS signals, control commands) should be encrypted using strong encryption protocols (e.g., AES-256 for data encryption, TLS for transmission); b) authentication – implement strong authentication mechanisms to ensure that only authorized users and devices can access and control the system; and c) data integrity – use cryptographic hashing (e.g., SHA-256) to ensure data integrity and detect any tampering with the transmitted data. Regarding data privacy, the Virtual Cockpit will perform sensitive data (e.g., identifiable GPS data) anonymization or pseudonymization to protect user privacy, while collecting and storing only the necessary data required for the system's functionality, avoiding excessive data collection that could lead to privacy issues. These actions will comply with EU data protection regulations that govern how personal data should be handled, stored, and processed. Although, apart from the prevention measures, the continuous monitoring of the system for any signs of security breaches or abnormal activity will be performed and a plan for incident response is being developed to quickly and effectively respond to security incidents or data breaches. By addressing these key security and data privacy concerns, the Virtual Cockpit can be made more secure and reliable, thereby protecting the integrity and privacy of the data while ensuring the safe and effective operation of the supported vehicles.

5 Conclusion

This deliverable has outlined the essential security and privacy activities for developing a secure data platform in the logistics and delivery sector. It has detailed the threat analysis, defined key security controls, and examined the implementation of key security technologies. Additionally, the deliverable introduced a VR-based interface for real-time interaction with smart vehicles, centralizing logistics operations. The focus was concentrated on the interface, which supports three primary data streams—live video, geolocation, and emergency commands—ensuring efficient communication. The second version of this deliverable (D4.6, to be released at M36, May 2026) will focus on the precise implementation details and necessary adaptations of these technologies to ensure seamless integration in the TRACE platform, while still addressing the challenges and complexities of a multi-stakeholder environment. The work on this deliverable and on the deliverable D3.1 “Report on reference architecture (A)” have been done in collaboration and have mutually influenced each other, as the architecture has been an input for the threat analysis, and security considerations have influenced the architecture itself. The output of threat analysis will be communicated to the demonstrator owners, to help them implement relevant security measures.

6 References

- [E. a. S. G. J. Levine, “RFC Style Guide,” 07 April 2021. [Online]. Available:
1 <https://www.ietf.org/archive/id/draft-flanagan-7322bis-07.html#:~:text=The%20lowercase%20%22must%22%20indicates%20those%20changes%20that%20will,question%20whether%20the%20guidance%20may%20be%20applied.%20%C2%B6>.
]]
- [Linaro Limited, “MCUboot,” Linaro Limited, 2024. [Online]. Available:
2 <https://www.trustedfirmware.org/projects/mcuboot/index.html>. [Accessed 2024 10 01].
]]
- [H2020 , *Autonomous Wireless Current Sensor for Aircraft Power Lines (785495)*., H2020-CS2-CFP06-3 2017-01.
]]
- [H2020, *An integrated emergency response decision support system for enhancing workers’ safety in offshore oil & gas operations (878950)*, H2020-EIC-FTI-2018-2020.
]]
- [A. D. a. R. M. a. D. Vizár, “Cryptology ePrint Archive,” 2022. [Online]. Available:
5 <https://eprint.iacr.org/2022/1275>. [Accessed 2024].
]]
- [S. L. T. Krenn, “Basic Secret Sharing,” in *An Introduction to Secret Sharing*, SpringerBriefs in Information
6 Security and Cryptography, 2023.
]]
- [L. Rotem, “Topics in Cryptography (CS 355, Lecture #3,” Stanford University, 2023.
7
]]

