Multi-layered Security Technologies

for hyper-connected smart cities

D3.5: Risks and security elements for a hyper-connected smart city

June 2020

Grant Agreement No. 814917

Multi-layered Security technologies to ensure hyper-connected smart cities with Blockchain, BigData, Cloud and IoT

Project acronym	M-Sec
Deliverable	D3.5 Risks and security elements for a hyper-connected smart city
Work Package	WP3
Submission date	June 2020
Deliverable lead	TST/KEIO
Authors	WLI, ICCS, CEA, AYTOSAN, NII, YNU, WU
Internal reviewer	ICCS/NII
Dissemination Level	Public
Type of deliverable	R
Version history	- V01, 27/May/2019, Arturo Medela, Jin Nakazawa, Initial ToC - V02, 30/January/2020, Antonis Litke, Arturo Medela, ToC Reviewed -V03-04, 05/February/2020, Arturo Medela, Jin Nakazawa, Final ToC - V1.0, 25/February/2020, Arturo Medela, Sections 1 and 2.1 - V1.1, 28/February, 2020, WLI, NII - V1.2, 31/March, 2020 AYTOSAN, GDPR - V1.3, 03/April, TST, Section 2.5, Section 3.1 - V1.4, 20/April, Keio - V1.5, 06/05, WLI, Xavier Cases, Sections 2.5 and 4 - V1.6, 16/06, TST, Sections 2.3, 2.4, 3.2 - V1.7, 18/06/2020, TST, Tables in Sections 2 and 4, Glossary update - V1.8, 19/06/2020, TST, ICCS, Tables updated, Executive Summary, new content in sections 2, 3 and 4 - V1.9, 22/06/2020, YNU, Commented or corrected all sections, except privacy - V2.0, 23/06/2020, TST, Editing prior internal review - V2.1, 28/06/2020, NII, Internal review - V2.2, 29/06/2020, ICCS, Internal review - V2.3, 30/06/2020, TST, review comments solved. Final editing.

The M-Sec project is jointly funded by the European Union's Horizon 2020 research and innovation programme (contract No 814917) and by the Commissioned Research of National Institute of Information and Communications Technology (NICT), JAPAN (contract No. 19501).

Table of Contents

List of Tables

List of Figures

Glossary

AMI	Access My Info
API	Application Program Interface
B2B	Business-to-Business
B2C	Business-to-Consumer
CIS	Center for Internet Security
CVE	Common Vulnerabilities and Exposures
CWE	Common Weakness Enumeration
ECDSA	Elliptic Curve Digital Signature Algorithm
EnCoRe	Ensuring Consent and Revocation
ENISA	European Union Agency for Network and Information Security
E-P3P	Platform for Enterprise Privacy Practices
FG	Functional Group
FMEA	Failure Mode and Effect Analysis
GCR	Governance, Compliance and Risk
HQL	Hibernate Query Language
HSTS	HTTP Strict Transport Security
JWT	JSON Web Token
KPI	Key Performance Indicator
LDAP	Lightweight Directory Access Protocol
LFI	Local File Inclusion
NIST	National Insitute of Standars and Technology
NoSQL	non-SQL
NVD	National Vulnerability Database
OCTAVE	Operationally Critical Threat, Asset, and Vulnerability Evaluation
OS	Operating System
OWASP	Open Web Application Security Project
PASTA	Process for Attack Simulation and Threat Analysis
PCI DSS	Payment Card Industry Data Security Standard
PET	Privacy Enhancing Technologies

PPL	PrimeLife Policy Language
RFI	Remote File Inclusion
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege
TEE	Trusted Execution Environment
TOTP	Time-based One-time Password
URI	Uniform Resource Identifier
XAML	eXtensible Application Markup Language
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language
XSS	Cross-Site Scripting
XXE	XML External Entities

Executive Summary

The work described in this deliverable (D3.5) was carried out in the framework of "WP3 – Requirements, architecture, hyper-connected smart city". The report presents the description of the risks that may affect the elements involved in such a smart city and sketches the security elements apt to ensure those risks do not generate a negative impact.

The reader is presented with detailed information on the risk assessment principles and procedures followed, as well as the underlying data and the literature references.

The structure of this document tries to align with the one presented in the rest of WP3 reports emanating from T3.1 and T3.2. In Section 1, an introduction explaining briefly the methodology followed is being presented. Next, in Sections 2, 3, and 4, each step in the risk analysis and mitigation approach is being presented, with the methodology and the results of each specific step being documented in detail. Finally, Section 5 concludes the deliverable.

The document is accompanied by a spreadsheet which has been the main source for extracting the main lists of threats (split into categories, groups, etc.), as well as the primary working file which was used internally by the consortium to gather and extract details for each threat. Some brief details about the usage of the spreadsheet are provided in the Annex section of this document.

All technical partners involved in this task collaborated and provided input regarding the potential threats looming over their contributions and thus over the M-Sec platform as a whole, and also provided some hints on the mitigation measures which will help to shape the project's novel Security layer in IoT contexts that will help to fulfil the fixed objectives.

This activity establishes a direct link with WP2 and WP4 activities and complements the work carried out in T5.3.

1. Introduction

1.1 Scope of the document

The current document is the deliverable “D3.5 - Risks and security elements for a hyper-connected smart city” which comprises the major outcome of “Task 3.3 - Risks and security elements for a hyper-connected smart city”. Task 3.3 is focused on identifying the elements and components necessary for a hyper-connected smart city.

In one part, the task specifically identifies existing elements and components that can be leveraged for the purpose. Such elements and components typically include, for example and not limited to, edge-side/cloud-side/end-to-end data delivery that need to be secured by M-Sec. In the other part, this task clarifies elements and components that must be addressed by M-Sec itself, and defines their requirements.

The primary audience of this document consists of the members of the consortium that participate in the design and development of the components and modules of the M-Sec system, as well as of the system per se. Additionally, the document is of wider interest to stakeholders that are active in the domain of smart cities, IoT, security and Big Data including researchers participating and contributing to H2020 projects under aforementioned topics.

1.2 Relation to other WPs and Tasks

The project as a whole has been working on clarifying potential risks through the use cases analysis and requirements analysis. The following Figure 1 gives an overview of the relations of this deliverable and the corresponding Task (T3.3) to other WPs, Tasks and deliverables.

Task 3.3 receives input from WP2 “Task2.1 - Use cases description” and “T2.2 - Pilots: Definition, setup and citizens involvement” through the corresponding deliverables (D2.1 and D2.2). More specifically, these deliverables provide an overview of the use cases description that in particular illustrates typical risks inherent in a hyper-connected smart city. Such risks are investigated in the Task 3.3 in an holistic way to highlight the security requirements for M-Sec components. It is also related to input from WP3 “Task 3.1 - System level and User level Requirements analysis” regarding requirements that reflect real citizen’s needs (D3.1). Among the requirements pointed out in D3.1, this document handles security requirements to consolidate a full set of security risks.

Regarding the exploitation of Task 3.3, having as an aim the full definition and consolidation of information related to M-Sec security requirements, the task provides its results to tasks included in “WP4 -Multi-layered Security Technologies”, which provides components included in the M-Sec architecture. Threat and Risk assessment will thus be completed before the finalisation of the development activities in WP4. This way, the risks shown in this document are mitigated by one or more components provided in WP4.

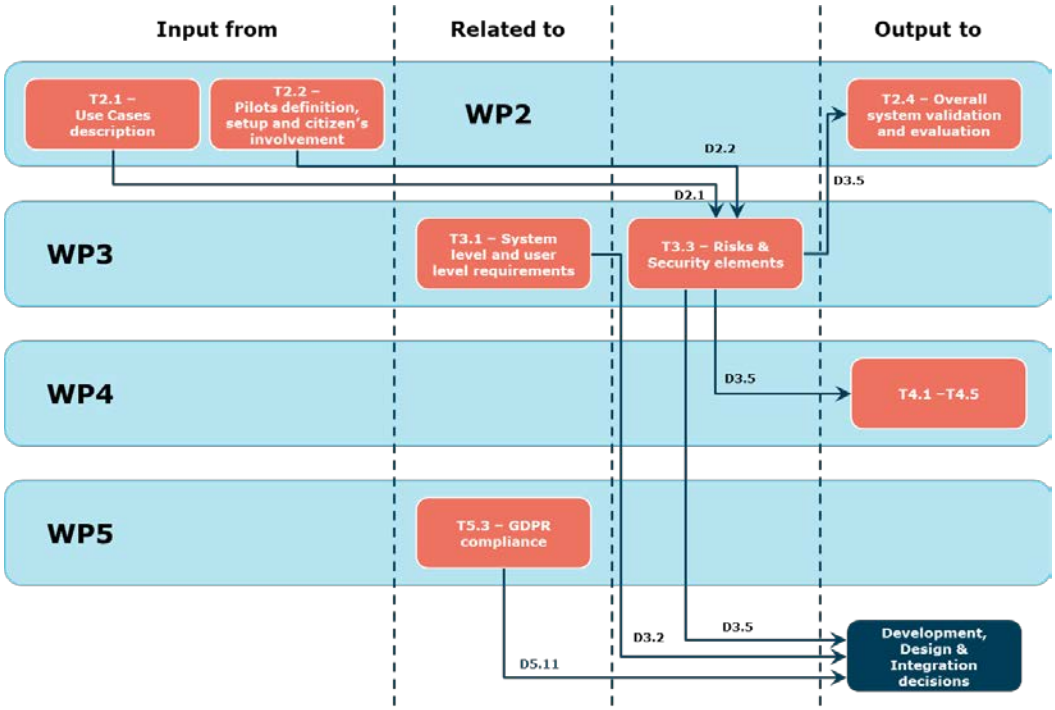

Figure 1. Relation of T3.3 and D3.5 to other WPs and Tasks

1.3 Document Structure & Methodology

The document starts in Section 2 by introducing various threat models for the smart city context and then selecting an appropriate one for M-Sec with which the risks and attacks are identified and a mitigation plan is proposed. The threat model proposed reflects on the attack surface for system/components in various layers, from the IoT layer to the Cloud and to to Application layer.

Afterwards, Section 3 provides a list of the methods for protecting data we are going to apply within M-Sec, starting with a quick recap of the existing Privacy Enhancing Technologies (PETs) and the observed trends. Overall, this will help to make decisions about M-Sec PETs of choice.

Next, Section 4 delves into risks based on NIST standards and guidance, starting with the methodology to proceed with their extraction and the way to do so from the M-Sec use cases. An exercise to categorize those follows and the section wraps with suggestions to proceed with their mitigation.

Finally, Section 5 summarizes the report and offers the main conclusions derived.

2. M-Sec Threat Models

Threat modelling works to identify, communicate, and understand threats and attack surface, for which mitigation controls can be designed within the context of protecting something of value.

Threat modelling can be applied to a wide range of things, including software, applications, devices, systems, networks, distributed systems, things in the Internet of Things, business processes, etc. There are very few technical products which cannot be threat-modelled. Threat modelling can be done at any stage of development, preferably early - so that security-by-design can be incorporated accordingly.

2.1 Threat Modelling Techniques

Threat modelling is a process by which potential threats, such as structural vulnerabilities or the absence of appropriate safeguards, can be identified and enumerated, and mitigations can be prioritized. The purpose of threat modelling is to provide defenders with a systematic analysis of what controls or defences need to be included, given the nature of the system, the probable attacker's profile, the most likely attack vectors, and the assets most desired by an attacker. Threat modelling answers questions like *'Where am I most vulnerable to an attack?'*, *'What are the most relevant threats?'*, and *'What do I need to do to safeguard against these threats?'*.

Conceptually, a threat modelling practice flows from a methodology. Numerous threat modelling methodologies are available for implementation. Typically, threat modelling has been implemented using one of four approaches independently, asset-centric, attacker-centric, and software-centric. Based on volume of published online content, the methodologies discussed below are the most well-known ones.

STRIDE

STRIDE is a model of threats developed by Praerit Garg and Loren Kohnfelder at Microsoft[1] for identifying system or computer security threats. It provides a mnemonic for security threats in six categories.

The threats are:

- **S**poofing
- **T**ampering
- **R**epudiation
- **I**nformation disclosure (privacy breach or data leak)
- **D**enial of service
- **E**levation of privilege

The STRIDE model was initially created as part of the process of threat modelling. STRIDE is a model of threats, used to help reason on and find threats to a system. It is used in conjunction with a model of the

[1] "*The Threats to Our Products*", Adam Shostack, Microsoft, https://www.microsoft.com/security/blog/2009/08/27/the-threats-to-our-products/

target system that can be constructed in parallel. This includes a full breakdown of processes, data stores, data flows and trust boundaries.

Today it is often used by security experts to help answer the question *"what can go wrong in this system we are working on?"*

Each threat is a violation of a desirable property for a system, as shown in Table 1:

Table 1. STRIDE model threat categories

Threat	Property Violated	Threat Definition
Spoofing	Authenticity	Pretending to be something or someone other than yourself
Tampering	Integrity	Modifying something on disk, network, memory or elsewhere
Repudiation	Non-repudiation	Claiming that you didn't do something or were not responsible for something; can be honest or false
Information disclosure	Confidentiality	Providing information to someone not authorized to access it
Denial of Service	Availability	Exhausting resources needed to provide service
Elevation of privilege	Authorization	Allowing someone to do something they are not authorized to do

Even though today there are several options on the table, STRIDE is still an effective tool to thoroughly understanding, *'What threats could this system or its components potentially experience in our production environment'*.

PASTA

The Process for Attack Simulation and Threat Analysis (PASTA)[2] is a risk-centric threat-modelling framework developed in 2012. It contains seven stages, each with multiple activities, which are illustrated in Table 2:

Table 2: PASTA threat model stages

PASTA model stages	
1. Define Objectives	Identify Business Objectives Identify Security and Compliance Requirements Business Impact Analysis
2. Define Technical Scope	Capture the Boundaries of the Technical Environment

[2] PASTA Process for Attack Simulation and threat analysis (PASTA) Risk-centric Threat Modeling, http://securesoftware.blogspot.com/2012/09/rebooting-software-security.html

	Capture Infrastructure | Application | Software Dependencies
3. Application Decomposition	Identify Use Cases | Define App. Entry Points & Trust Levels Identify Actors | Assets | Services | Roles | Data Sources Data Flow Diagramming (DFDs) | Trust Boundaries
4. Threat Analysis	Probabilistic Attack Scenarios Analysis Regression Analysis on Security Events Threat Intelligence Correlation and Analytics
5. Vulnerability & Weaknesses Analysis	Queries of Existing Vulnerability Reports & Issues Tracking Threat to Existing Vulnerability Mapping Using Threat Trees Design Flaw Analysis Using Use and Abuse Cases Scorings (CVSS/CWSS) | Enumerations (CWE/CVE)
6. Attack Modelling	Attack Surface Analysis Attack Tree Development | Attack Library Mgt. Attack to Vulnerability & Exploit Analysis Using Attack Trees
7. Risk & Impact Analysis	Qualify & Quantify Business Impact Countermeasure Identification and Residual Risk Analysis ID Risk Mitigation Strategies

Attack Trees

Using attack trees to model threats is one of the oldest and most widely applied techniques on cyber-only systems, cyber-physical systems, and purely physical systems. Attack trees were initially applied as a stand-alone method and have since been combined with other methods and frameworks.

Attack trees are diagrams that depict attacks on a system in tree form. The tree root is the goal for the attack, and the leaves are ways to achieve that goal. Each goal is represented as a separate tree. Thus, the system threat analysis produces a set of attack trees. See examples in Figure 2.

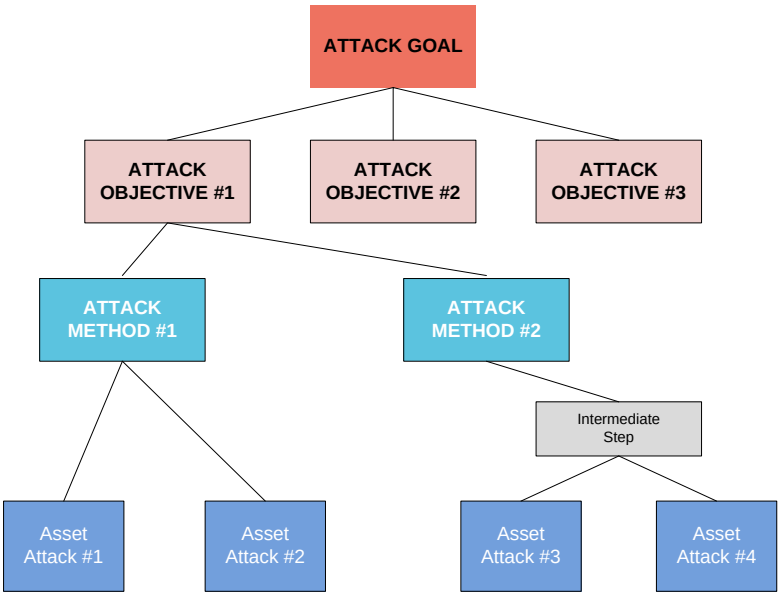

Figure 2: Attack Tree example

In the case of a complex system, attack trees can be built for each component instead of for the whole system. Administrators can build attack trees and use them to inform security decisions, to determine whether the systems are vulnerable to an attack, and to evaluate a specific type of attack.

In recent years, this method has often been used in combination with other techniques and within frameworks such as STRIDE, CVSS, and PASTA.

OCTAVE

The Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) approach defines a risk-based strategic assessment and planning technique for security. OCTAVE is a self-directed approach, meaning that people from an organization assume responsibility for setting the organization's security strategy. The team draws on the knowledge of many employees to define the current state of security, identify risks to critical assets, and set a security strategy. It can be tailored for most organizations.

Unlike most other risk assessment methods, the OCTAVE approach is driven by operational risk and security practices and not technology. It is designed to allow an organization to:

- Direct and manage information security risk assessments for themselves
- Make the best decisions based on their unique risks
- Focus on protecting key information assets
- Effectively communicate key security information

OCTAVE has three phases:

1. Build asset-based threat profiles. (This is an organizational evaluation.)
2. Identify infrastructure vulnerability. (This is an evaluation of the information infrastructure.)
3. Develop a security strategy and plans. (This is an identification of risks to the organization's critical assets and decision making.)

OCTAVE-S is a variation of the approach tailored to the limited means and unique constraints typically found in small organizations (less than 100 people). OCTAVE-S is led by a small, interdisciplinary team (three to five people) of an organization's personnel who gather and analyse information, producing a protection strategy and mitigation plans based on the organization's unique operational security risks. To conduct OCTAVE-S effectively, the team must have broad knowledge of the organization's business and security processes, so it will be able to conduct all activities by itself.

2.2 M-Sec Threat Model Definition

An IoT system such as the one discussed by M-Sec includes different pieces of devices and software, ranging from IoT devices (e.g. sensors) and cloud servers, to user-side mobile devices. There is a need to conduct a multi-dimensional threat and risk assessment that can provide solid ground for multi-layered security in M-Sec. Therefore, we have selected STRIDE threat model for fulfilling this need of M-Sec Project. The STRIDE threat model is closer to systems and related components that are being developed in M-Sec architecture and has been used in another similar development project by consortium partners; therefore, M-Sec partners decided to use STRIDE for evaluating threats against each entry point. This will help to identify, assess, and classify potential weak areas or threat vectors and risks more granularly with respect to M-Sec IoT, cloud, and application layers

In order to optimize security best practices, it is recommended that a typical IoT architecture such as the one represented by the M-Sec framework is divided into several component/zones as part of the threat modelling exercise, aligned with the layers in the M-Sec architecture (namely IoT, Middleware, Cloud, and Application). These zones are described fully throughout this document and include:

- IoT (Devices in the Edge and Gateways),
- Communication (IoT to Cloud, Blockchain, …)
- Cloud,
- Applications

These are complemented by the end-to-end (covering the whole path the information travels through, from its generation until reaching the end user) related threats and the ones strictly associated to Privacy.

This Threat Modelling activity offers the greatest value when incorporated into the design phase, since it provides greater flexibility to make changes to eliminate threats. If not, mitigations will be added, while testing them and ensuring they remain current for the long term.

When considering the threat modelling, it is convenient to consider the solution as a whole and also focus on the following areas:

- The security and privacy features.
- The features whose failures are security relevant.
- The features that touch a trust boundary.

The subsequent threat modelling process will be composed of four steps, namely:

- Model the system
- Enumerate Threats
- Mitigate Risks

- Validate those mitigations

The threat model used in M-Sec is defined as encompassing following four core elements:

- Processes such as web services, Win32 services, and *nix daemons. Some complex entities (for example field gateways and sensors) can be abstracted as a process as well when no other alternative is possible.
- Data stores: anywhere data is stored, such as a configuration file or database.
- Data flow: where data moves between other elements in the system.
- External Entities: anything that interacts with the system, but is not under the control of the application, examples include users and satellite feeds.

2.3 Risk rating criteria

Taking as a basis the Risk Management Guide for Information Technology Systems published by NIST (SP 800-30)[3], M-Sec takes into consideration the concepts reflected in Table 3 to establish the Risk Criteria to be followed in the project.

Table 3. Risk Management Guide followed in M-Sec

Terms	Definitions
Confidentiality	Protection of information from unauthorized, unanticipated, or unintentional disclosure which could result in loss of trust or public confidence.
Integrity	Refers to the requirement that information must be protected from improper modification. Integrity is lost if unauthorized changes are made to the data or IT system by either intentional or accidental acts. If the loss of system or data integrity is not corrected, continued use of the contaminated system or corrupted data could result in inaccuracy, fraud, or erroneous decisions.
Availability	Loss of component/system functionality and operational effectiveness that can prevent intended services.
Vulnerability	A flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited) and result in a security breach or violation of the system's security policy.
Threat	The potential for a threat-source to exercise (accidentally trigger or intentionally exploit) a specific vulnerability.
Likelihood	The possibility that a given event will occur. Terms sometimes take on more specific connotations, with "likelihood" indicating the possibility that a given event will occur in qualitative terms such as high, medium, and low, or other judgmental scales, and "probability" indicating a quantitative measure such as a percentage, frequency of

[3] NIST, https://www.nist.gov/publications/risk-management-guide-information-technology-systems

Terms	Definitions
	occurrence, or other numerical metric.
Impact	Result or effect of an event. There may be a range of possible impacts associated with an event. The impact of an event can be positive or negative relative to the intended objectives.
Risk	The possibility that an event will occur and adversely affect the achievement of M-Sec objectives.

The first step in the Risk Evaluation process consists of assigning a certain probability to the possible appearance of a certain risk that may affect some M-Sec assets. This will be understood as a 3-tiered classification, as shown in Table 4.

Table 4. Probability Score

PROBABILITY		
PROBABILITY SCORE	**DESCRIPTOR**	**FREQUENCY**
1	**Unlikely**	Do not expect it to happen/recur, but it is possible it may do so
3	**Likely**	Will probably happen/recur, but it is not a persisting issue/circumstances
5	**Almost certain**	Will undoubtedly happen/recur, possibly frequently

Afterwards, it is turn to evaluate how critical it would be for that M-Sec asset the real appearance of this threat and its envisioned impact over the M-Sec infrastructure. A similar ranking is employed, and its description is given in Table 5.

Table 5. Criticality Score

CRITICALITY		
CRITICALITY SCORE	**DESCRIPTOR**	**DEFINITION**
1	**LOW**	Component/System/Application capabilities are **not critical** for M-Sec Pilot. Information affected may only contain small number of useful information. Compromise of this function/data would result in **negligible adverse impact** to M-Sec Pilot or its partners/participents resulting from confidentiality, integrity, or availability of the function/data being compromised. o **Confidentiality:** Incidental or non-critical data exposure. No reputation impact. o **Integrity:** Critical data cannot be altered. For non-critical information, in the event of alteration or failure, it would have a minimal impact or could be replaced with minimal staff time or expenses. o **Availability:** In the event of component/system failure, it would have a minimal impact on operations or services, and could be fixed with minimal staff time or expense.
3	**MEDIUM**	Component/System/Application capabilities are **important** but **not critical** for M-Sec Pilot. Information affected is important to M-Sec Pilot/Partners and should be accessed only by a limited group of people. Information should be reasonably protected against disclosure to unauthorized users and/or intentional acts that are considered to be malicious and/or destructive. Compromise of this function/data would result in **limited adverse impact** to M-Sec Pilot or its partners/participents resulting from confidentiality, integrity, or availability of the function/data being compromised. o **Confidentiality:** Incidental or limited critical data exposure. Limited reputation impact. o **Integrity:** Possibility of critical data alteration. o **Availability:** In the event of component/system failure, it would have a limited impact on operations or services, and could be fixed with moderate staff time or expense.
5	**HIGH**	Component/System/Application capabilities are **important and critical** for M-Sec Pilot. Information affected is confidential to M-Sec Pilot/Partners and should be accessed only by specific authorized people. Loss, compromise, or unauthorized modification of the information asset (e.g., process, data, or application) could result in major operational/data loss during M-Sec Pilot. Information should be properly protected against disclosure to unauthorized users and/or intentional acts that are considered to be malicious and/or destructive. Compromise of this function/data would result in **major adverse impact** to M-Sec Pilot or its partners/participents resulting from confidentiality, integrity, or availability of the function/data being compromised. o **Confidentiality:** Alteration, destruction or exposure of information would have significant harm to M-Sec Pilot objectives. Major reputation impact. o **Integrity:** Certainity of critical data alteration. o **Availability:** In the event of component/system failure, it would have a High impact on operations or services, and could be fixed with major staff time or expense.

Upon collecting this information, it is possible to create some so-called Risk Bands. Risk scores can be calculated by multiplying the two values above together, giving a maximum possible score of 25.

Risk Rating = (Probability Score) x (Criticality Score)

By grouping Risk scores into bands and colour coding them, as in the example below, we get the so called Risk Bands. Red is typically used for high risks and Orange represents medium risks. For such risks, the risk criteria mandates that action that should be taken to lower the risks. Green is usually used for lower risks that may not require immediate action or may be acceptable risks.

The obtained result will help with prioritizations in designing the relevant security plan depending on the Risk Rating value, stating whether that particular Risk requires immediate action or not (see Table 5).

Table 6. Risk Rating and required actions

RISK RATING	RISK	ACTION REQUIRED
1 - 3	LOW	ACCEPT RISK Manage by routine procedures and existing policies/procedures
4 - 15	MEDIUM	ACTION REQUIRED
16 - 25	HIGH	IMMEDIATE ACTION REQUIRED

A visual representation of these Risk Bands is depicted in Figure 3.

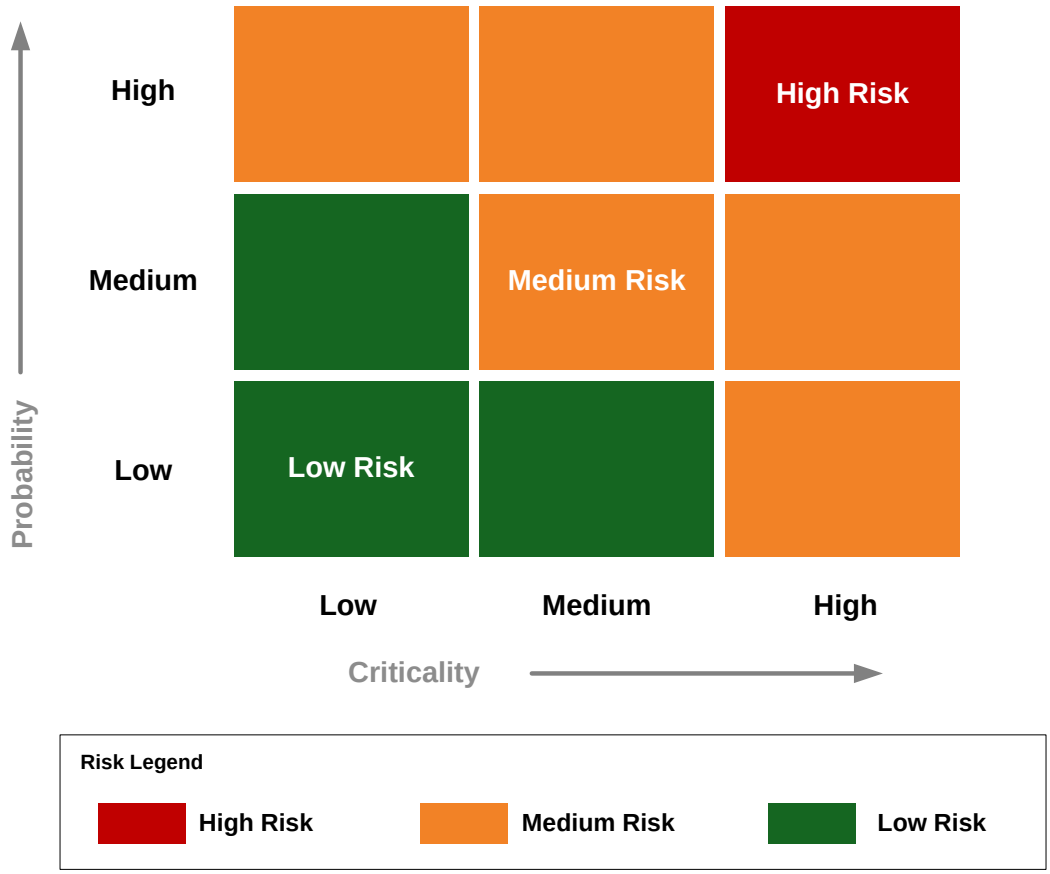

Figure 3. Risk Bands

As mentioned, M-Sec conducts this risk categorization exercise taking into account the different layers of its architecture and assigning the corresponding Risk Ratings to a long list of potential threats that could affect the proper functioning of the project demonstrators.

2.4 Risks categorization

The risks to M-Sec architecture will be categorized into IoT (edge), communication, cloud, and application parts, establishing a certain link to Work Package 4 and its deliverable. They all will follow the STRIDE guidelines. Identifying the security risk area each of them affects to and specifying the information security attribute that particular threat points to, according to the qualities desirables for Information Systems of Confidentiality, Integrity and Availability (CIA). The match among the different categories in the STRIDE model with the Security Risk Areas that will be affected and the Impact they could cause is summarized in Table 7 below.

Table 7. Link of Security Risk Areas and Impacts to STRIDE Threats

Threats	Security Risk Area	Impacts (CIA)
Spoofing	Authentication	Integrity
Tampering	Integrity	Integrity
Repudiation	Non-repudiability	Integrity
Information disclosure	Confidentiality	Confidentiality
Denial of Service	Availability	Availability
Elevation of Privilege	Authorization	Integrity

It is worth noting that during this risk categorization process there will also be references to certain threats related to privacy that will be a matter of further discussion in Task 5.3 and Deliverable 5.11.

IoT Risks

The main focus of this task related to the implementation of the IoT security framework within M-Sec, consists in helping to develop reliable and secure applications for the Smart City context. The goal here consists in looking for techniques, methods, and design and operating principles that minimize the risk of suffering critical vulnerabilities in a wide range of IoT devices, which could be leveraged by hackers to carry out a number of nefarious activities.

To properly provide a detailed list of threats that may affect the IoT devices which will be part of the M-Sec use cases there is a depiction of the interfaces that will be involved in the information exchanges involving those IoT devices. Figure 4 presents this sketch where readers can see the interfaces that will be mentioned in the corresponding Risk evaluation.

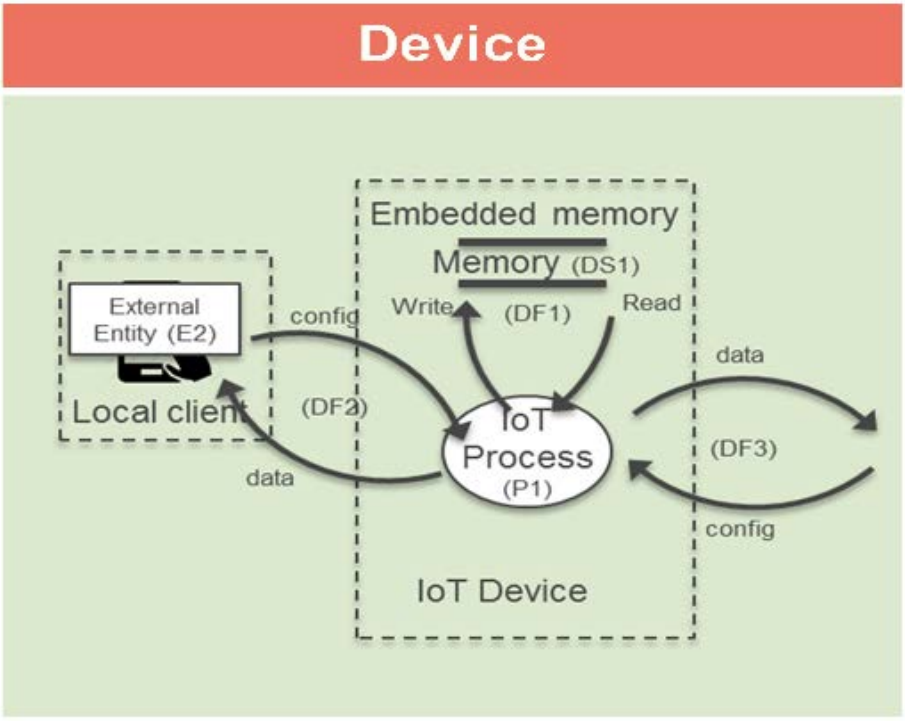

Figure 4. Interfaces for the IoT devices

This reference helps to better understand the exercise reflected in Table 8 where the relevant threats affecting IoT devices are presented, along with the risk rating estimated for each one of them. Readers might find some threats present no associated risk rating; this situation will be further addressed in Section 4.

Table 8. M-Sec IoT related threats

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RISK RATING
Thr.IoT.1	Data stored in the device can be read by an intruder	I	P1, DF1, DS1	EnMon, Crow, Caburn	Use Case 1, 2	3	3	9
Thr.IoT.2	An unauthorized party can modify data on the device	T	P1, DF1, DS1	EnMon, Crow, Caburn	Use Case 1, 2	3	3	9
Thr.IoT.3	Man in the middle attack: a third party puts itself between the entity that communicates with the device and the device itself, without them noticing (so that both actually communicate with the intruder)	S	DF2, DF3	EnMon, Crow, Caburn	Use Case 1, 2	3	3	9
		S	DF2, DF3	KEIO Mobile Sensing Platform	Use Case 3	3	3	9
Thr.IoT.4	Unauthorized modification of configuration parameters of the device or the sensor	E, T, D	P1, DS1	EnMon, Crow, Caburn	Use Case 1, 2	3	3	9
		E, T, D	Sensor/P1, DS1	KEIO Mobile Sensing Platform	Use Case 3	1	3	3
Thr.IoT.5	An attacker can overload the device by injecting many requests	D	N/A	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	Use Case 1, 2, 3	-	-	-

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RISK RATING
Thr.IoT.6	Jamming of the wireless communication link	D	N/A	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	Use Case 1, 2, 3	-	-	-
Thr.IoT.7	Accidental or intentional physical damage to any device part may cause device failure	D	Device HW	EnMon, Crow, Caburn	Use Case 1, 2	3	3	9
		D	Device HW	KEIO Mobile Sensing Platform	Use Case 3	3	3	9
Thr.IoT.8	Insecure firmware update mechanism: the firmware has been retrieved at a non-valid source	S	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-
Thr.IoT.9	If installed, malware has full access to the whole device	T	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-
Thr.IoT.10	If installed, malware has access to data in the device	I, T	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RISK RATING
Thr.IoT.11	If installed, the malicious firmware may cause device operation failure	D	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-
Thr.IoT.12	Insecure firmware update mechanism: the firmware is corrupted	D	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-
Thr.IoT.13	A device designed to be used by several users and keeping history per user discloses information on the other users (spontaneously or after wrong manipulation).	I, E	N/A	EnMon, Crow, KEIO Mobile Sensing Platform	Use Case 1,3	-	-	-
Thr.IoT.14	A weak authentication method is very likely to be used (short and simple passwords, if any), opening a door to data and device exposure (breaches to integrity and confidentiality).	S, Implementation issue	N/A	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	Use Case 1, 2, 3	-	-	-
Thr.IoT.15	The device was reset to its default settings, which does not include security.	E	N/A	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	Use Case 1, 2, 3	-	-	-

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RISK RATING
Thr.IoT.16	Nobody is responsible for device maintenance.	Management issue	Life Cycle	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	Use Case 1, 2, 3	1	3	3
Thr.IoT.17	Nobody is responsible for system management and maintenance (e.g. system: device network)	Management issue	Life Cycle	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	Use Case 1, 2, 3	1	3	3
Thr.IoT.18	Attack on Power Management …	D	Device HW	Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	Use Case 1,2,3	3	3	9
Thr.IoT.19	A visitor is playing with the device (e.g. a blood pressure monitor) and wrongly records measurements that are not those of the intended user.	S: Identification rather than authenticatio n	P1, DF1, DS1	Caburn IoT Devices	Use Case 2	3	5	15

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RISK RATING
Thr.IoT.20	Old persons may not know how to handle electronic devices efficiently. They make errors and quit easily.	Usability	P1, DF1, DS1	Caburn IoT Devices	Use Case 2	3	5	15
Thr.IoT.21	An old person denies having recorded a measurement (e.g. blood pressure rate).	R	P1, DF1, DS1	Caburn IoT Devices	Use Case 2	3	3	9
Thr.IoT.22	A member of caring personnel denies having administered a treatment.	R	N/A	N/A	N/A	--	--	--

Communication Level Risks

To properly provide a detailed list of threats that may act at Communication level there is a depiction of the interfaces that will be involved in the related information exchanges (see Figure 5)

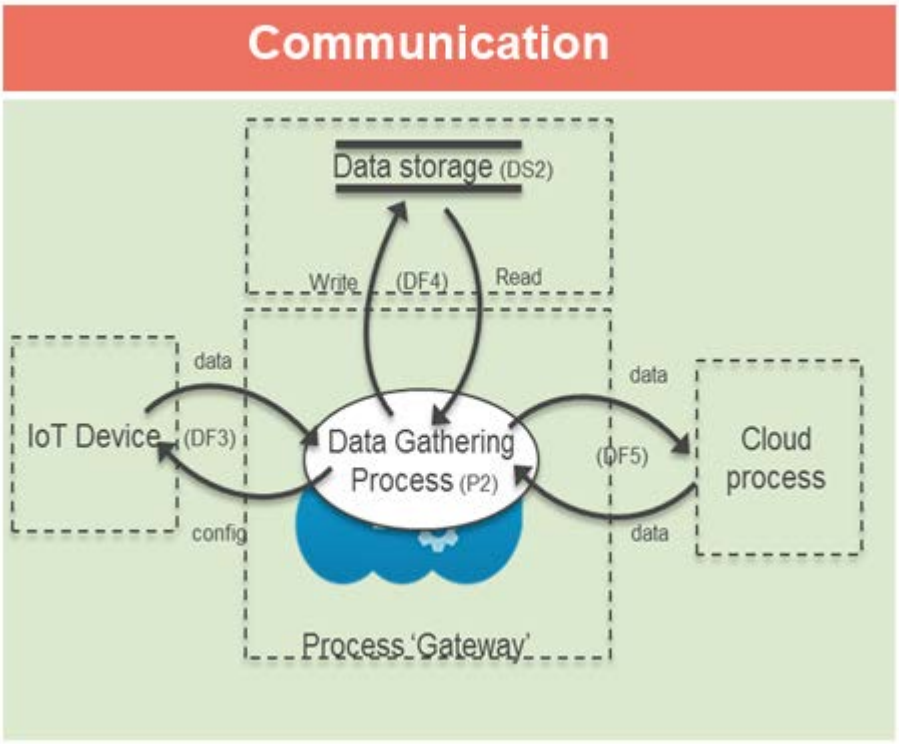

Figure 5. Interfaces for the Gateways

This reference helps to better understand the exercise reflected in Table 9 which presents the relevant threats affecting Communication layer of the M-Sec framework, along with their associated risk rating.

Nevertheless, before going into that table, it is worth mentioning the **Blockchain level risks** which may arise within this context. Blockchains fall under two types: permissionless and permissioned chains. Permissionless blockchains allow any party to participate in the network, while permissioned blockchains are formed by organizations or administrators who evaluate the participation of an entity on the blockchain based on their own procedures. In permissionless blockchains anyone can participate in the blockchain validation process (e.g. as a block miner as long as they meet certain technological requirements dictated by the network. No other entity checks, such as know your customer (KYC) or other background checks of the service provider, are possible in these blockchains. Additionally, permissionless blockchains have privacy issues that pose a significant risk. In permissioned blockchains the consortium network or the administrator can predefine the update process. Usually, this involves a choice of a consensus algorithm that is deployed on the network to update the blockchain ledger. Privacy issues can be handled by the choice of infrastructure by the participants, and suspicious activity monitoring can be deployed across the network by the administrator or the consortium. Therefore, this framework is more suitable for institutions to use with a group of known and predetermined peers.

There are several cryptographic protocols that are used to achieve consensus among participant nodes for updating the blockchain ledger. This applies to both permissioned and permissionless blockchains. In the former ones it is usually a delegated / voting mechanism which is cost effective while in the latter expensive (power consuming) computational algorithmic approaches have to be applied.

The 51% attack risk: The blockchain relies on the distributed consensus mechanism to establish mutual trust. However, the consensus mechanism itself has 51% vulnerability, which can be exploited by attackers to control the entire blockchain. More precisely, in Proof of Work based blockchains, if a single miner's hashing power accounts for more than 50% of the total hashing power of the entire blockchain, then the 51% attack may be launched. In Proof of Stake based blockchains, 51% attack may also occur if the number of coins/tokens owned by a single entity is more than 50% of the total blockchain. By launching the 51% attack, an attacker can arbitrarily manipulate and modify the blockchain information. Specifically, an attacker can exploit this vulnerability to conduct the following attacks: (1) Reverse transactions and initiate double spending attack (the same coins are spent multiple times); (2) Exclude and modify the ordering of transactions; (3) Impede the confirmation operation of normal transactions.

Private key security risk: In blockchains, the user's private key is regarded as the identity and security credential, which is generated and maintained by the user instead of third-party agencies. Various vulnerabilities have been discovered in the digital signature algorithmic schemes such as in ECDSA (Elliptic Curve Digital Signature Algorithm) scheme, through which an attacker can recover the user's private key because it does not generate enough randomness during the signature process. Once the user's private key is lost, it will not be able to be recovered. If the private key is stolen or revealed by a malicious user, the user's blockchain account will face the risk of being tampered by others. Since the blockchain is not dependent on any centralized third-party trusted institutions, if the user's private key is stolen, it is difficult to track the criminal's behaviours and recover the modified blockchain information.

Table 9. Communication related threats

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.Com.1	Communications may be eavesdropped	I	DF3, DF5	IoT Gateway, Caburn	Use Case 2, 3	3	5	15
Thr.Com.2	Unrestricted access to alter device configuration	T	DF3	IoT Gateway, Caburn	Use Case 2, 3	3	5	15
Thr.Com.3	Data storage is readable without authentication	I, E	DF4	IoT Gateway, EnMon, Crow, Caburn	Use Case 1, 2, 3	3	3	9
Thr.Com.4	Data storage is writeable without authentication	T, E	DF4	IoT Gateway, Caburn	Use Case 2, 3	3	3	9
Thr.Com.5	IoT physical interfaces (USB dongles, etc.) are removed	D	P2	IoT Gateway	Use Case 3	5	5	25
Thr.Com.6	Device is removed or put out of range	D	DF3, P2	IoT Gateway, EnMon, Crow, Caburn	Use Case 1, 2, 3	3	5	15
Thr.Com.7	Gateway is unplugged to free a power plug	D	P2	IoT Gateway, Caburn	Use Case 2, 3	5	5	25
Thr.Com.8	Gateway is stolen for reverse engineering	I, D	P2	IoT Gateway, EnMon, Crow, Caburn	Use Case 1, 2, 3	3	3	9

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.Com.9	Disclosure of encryption parameters for the communication channels	I	DF5	IoT Gateway, Caburn	Use Case 2, 3	3	5	15
Thr.Com.10	Attack on the security implementation	D, E	P2	IoT Gateway	Use Case 3	5	5	25
Thr.Com.11	Attack by replay or Man-in-Middle Attack	D, S	DF3, DF5	IoT Gateway, EnMon, Crow, Caburn	Use Case 1, 2, 3	3	5	15
Thr.Com.12	DHCP network parameters are compromised to spoof other processes (ARP, DNS)	S, T	DF5 (DF3)	IoT Gateway	Use Case 3	1	3	3
Thr.Com.13	Wireless signals are jammed (GPRS, WiFi, other)	D	DF5 (DF3)	IoT Gateway, Crow, Caburn	Use Case 1, 2, 3	1	3	3
Thr.Com.14	An attacker gains access to the boot process using a physical port (serial console, JTAG)	E	P2	IoT Gateway	Use Case 3	3	3	9
Thr.Com.15	An attacker gains access to the operating system using a service on the system (telnet, FTP, SSH, etc.)	E	P2	IoT Gateway, Caburn	Use Case 2, 3	3	5	15
Thr.Com.16	A weak authentication method (short/simple/default passwords, if any), opening a door to data and device exposure (breaches to integrity and	S, Implementation issue	P2	IoT Gateway, Caburn	Use Case 2, 3	3	5	15

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
	confidentiality).							
Thr.Com.17	If installed, malware has full access to the device or data	I, T	P2, DF4, DS2	IoT Gateway, Caburn	Use Case 2, 3	3	5	15
Thr.Com.18	Accidental or intentional physical damage to any device part may cause device failure	D	HW	IoT Gateway, EnMon, Crow, Caburn	Use Case 1, 2, 3	3	3	9
Thr.Com.19	The device was reset to its default settings, which does not include security.	E	P2, DF5	IoT Gateway, Caburn	Use Case 2, 3	5	5	25
Thr.Com.20	Nobody is responsible for device management and maintenance.	Management issue	Life cycle	IoT Gateway, EnMon, Crow	Use Case 1, 3	1	3	3
Thr.Com.21	51% attack over blockchain	R, T	System Level	Quorum Blockchain	All Use Cases	1	5	5
Thr.Com.22	Private key security - Public key encryption scheme	R, E	System Level	Quorum Blockchain	All Use Cases	1	3	3
Thr.Com.23	Data storage is physically removable	D	DS2, DF4	N/A	Use Case 3	-	-	-
Thr.Com.24	Removable data storage (SD card with OS) is stolen to extract security credentials	D	DS2	N/A	Use Case 3	-	-	-
Thr.Com.25	Over consumption of resources (processor, memory) due to an unmonitored activity	D	DS2, P2	N/A	Use Case 3	-	-	-

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.Com.26	Gateway OS is updated with a malicious update	S, E	P2	N/A	Use Case 3	-	-	-
Thr.Com.27	If installed, the malicious firmware may cause device operation failure	D	All	N/A	Use Case 3	-	-	-
Thr.Com.28	The boot process is altered to run a rogue OS	D, E	P2	N/A	Use Case 3	-	-	-
Thr.Com.29	The system is unupdated, leaking access to potential attackers	E	P2	N/A	Use Case 3	-	-	-
Thr.Com.30	Pairing the gateway with non-legitimate devices	S	DF3	N/A	Use Case 3	-	-	-

Cloud Level Risks

Increasing number of reported breaches and the fear of losing data are among some of the factors that makes the data owners very nervous. The confidentiality, integrity, availability, and privacy of data become questionable as many users have no insight into how their data is being protected by the cloud vendors. Therefore, understanding the risks to data in-motion and at-rest is critical for building user's trust in M-Sec platform.

To properly provide a detailed list of threats that may act at a Cloud level, where data is stored, we need to carefully look at each possible entry point or interfaces in our system that will be involved in the related information exchanges (see Figure 6)

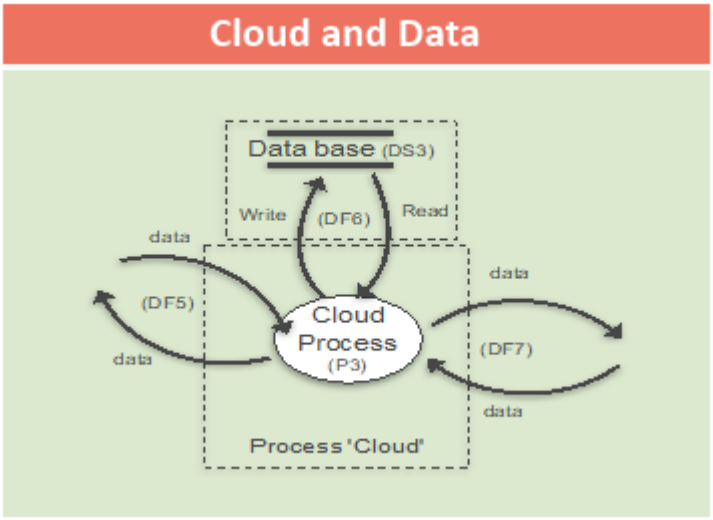

Figure 6. Interfaces for Cloud and Data

This reference helps to better understand the exercise reflected in Table 10 which presents the relevant threats affecting the Cloud layer of the M-Sec framework, where the Data Storage Functional Groups can be found, along with their associated risk rating.

Table 10. M-Sec Cloud related threats

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.CD.1	Impersonation: A third party uses a false ID to gain access to the cloud	S	P3	SoxFire	Use Case 3	3	3	9
Thr.CD.2	An attacker may install a malware to access the data and whole cloud system	I, T, D	P3, DS3	SoxFire	Use Case 3	1	3	3
Thr.CD.3	Accidental or intentional physical damage to any cloud part may cause cloud service failure	D	Cloud HW	SoxFire	Use Case 3	3	1	3
Thr.CD.4	Disruption of a global service (e.g. attack on power management)	D	Cloud HW, P3	SoxFire	Use Case 3	3	1	3
Thr.CD.5	Data (raw & processed, personal data) stored in the cloud can be read by an intruder	I	DS3	SoxFire	Use Case 3	3	5	15
Thr.CD.6	An unauthorized party gets access to device configuration information	I	DS3	SoxFire	Use Case 3	1	3	3
Thr.CD.7	Attacker denies legitimate users access to infrastructure services	D	P3	N/A	-	-	-	-
Thr.CD.8	Attacker can poison cloud database and/or alters outgoing information	T	DS3, DF7	SoxFire	Use Case 3	3	5	15
Thr.CD.9	An unauthorized party can modify data stored in the cloud data base	T	DS3, DF6	SoxFire	Use Case 3	1	5	5
Thr.CD.10	Disclosure of private services, access policies, cryptographic material, user information	I	DS3, P3	SoxFire	Use Case 3	1	5	5

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.CD.11	Attacker gains knowledge of sensitive exchange data	I	DF5, DF6, DF7	SoxFire	Use Case 3	3	5	15
Thr.CD.12	Alteration of the invocation of a service	T	DF5, DF6, DF7	SoxFire	Use Case 3	1	3	3
Thr.CD.13	Attacker can disrupt communications	D	DF6, DF7	SoxFire	Use Case 3	1	5	5
Thr.CD.14	Man in the middle attack: a third party puts itself between the entity (e.g. gateway or application) that communicates with the cloud and the cloud itself, without them noticing (so that both actually communicate with the intruder)	S	DF5, DF6, DF7	SoxFire	Use Case 3	3	5	15
Thr.CD.15	Wrong authorization information propagating from one server to another	E	DF6, DF7	N/A	-	-	-	-
Thr.CD.16	User is involved in transactions/actions with a malicious peer	R	P3, DF7	N/A	-	-	-	-
Thr.CD.17	A service critical for user's safety is disabled	D	DS3, P3, User	N/A	-	-	-	-
Thr.CD.18	Nobody is responsible for cloud maintenance	Management issue	Life Cycle	SoxFire	Use Case 3	3	3	9
Thr.CD.19	Nobody is responsible for system management and maintenance (e.g. cloud infrastructure)	Management issue	Life Cycle	SoxFire	Use Case 3	3	3	9

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE (Use Case #)	PROBABILITY	CRITICALITY	RISK RATING
Thr.CD.20	An old person denies having recorded a measurement (e.g. blood pressure rate)	R	System Level	N/A	-	-	-	-
Thr.CD.21	A member of caring personnel denies having administered a treatment	R	System Level	N/A	-	-	-	-
Thr.CD.22	Attacker changes the association between the device id / personal id and the corresponding device or an old person	S, T	System Level	N/A	-	-	-	-
Thr.CD.23	Attacker changes configurations and prevents proper communication to an actuator	D	System Level	SoxFire	Use Case 3	3	3	9

Application Level Risks

At the **application level** there is a clear need for secure application development, which has increased with the growing number of services on the Internet. It is required for secure application development to consider various types of security concerns.

As a first approach, the general risks listed in Table 11 are taken into consideration, and drive the initial evaluation of threats to M-Sec applications, paying special attention to Use Case 3, where even though no specific app will be exactly employed, users will be able to check results extracted from the dat collection side.

Table 11. Generic Application Level Risks related to UC3

Risks (CIA)	Threads (STRIDE)	Definition (STRIDE)	M-Sec misusecase example (Use Case 3)
Confidential	Spoofing	Impersonating something or someone else	Malicious attacker reveal environment data through IoT
	Information Disclosure	Exposing information to someone not authorized to see it	
	Elevation of Privilege	Gain capabilities without proper authorization	
Integrity	Tampering	Modifyng data or code	Malicious attacker consume unauthentic environment data through application
	Repudiation	Claiming to have not performed an action	
Availability	Denial of Service	Deny or degrade service to users	Malicious attacker interrupt that data supplier visualize environment data

The following list shows the found risks for the application level of security risks. If you want more information about application level risks, you can check some online resources from The Open Web Application Security Project (OWASP)[4], the Common Weakness Enumeration (CWE) from SANS[5], the National Insitute of Standars and Technology (NIST) from U.S Department of Commerce[6] and the Center for Internet Security (CIS)[7] .

Injection.

Injection flaws, such as SQL, NoSQL, OSCommand, Argument, Expression Language, LDAP and XML injection (aka Blind XPath Injection) occur when untrusted data is sent to an interpreter as part of a command or query because of an improper neutralization of Special Elements or Argument Delimiters. The

[4] https://owasp.org/
[5] https://www.sans.org/
[6] https://www.nist.gov/
[7] https://www.cisecurity.org/

attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

Broken Authentication.

Application functions related to authentication and session management, such as unprotected storage of credentials, insuficciently protected credentials, unprotected transport of credential, insufficient session expiration, unverified password change, weak password recovery mechanism and use opf single-factor authentication are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.

Sensitive Data Exposure.

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII by missing or inadequate encryption strength, cleartext transmission/storage, improper certificate validation or key maanagement errors amoung others. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

XML External Entities (XXE).

Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

Broken Access Control.

Restrictions on what authenticated users are allowed to do are often not properly enforced, it might be due to an improper authorization, an improper access control, an improper limitation of a pathname to a restricted directory, a direct request or an authorization bypass through user-controlled key. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

Security Misconfiguration.

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

Cross-Site Scripting XSS.

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Insecure Deserialization.

Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

Using Components with Known Vulnerabilities.

Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

Insufficient Logging & Monitoring.

Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

Smart contracts related Risks

Regardless of the type of blockchain, the business logic of the applications supported by the blockchains is encoded using smart contracts. Smart contracts are self-executing code on the blockchain framework that enable direct processing, which means that manual intervention is not required to execute transactions. Smart contracts rely on data from outside entities referred to as "*oracles*," and can act on data associated with any public address or with another smart contract on the blockchain. A smart contract can mimic a contract and can execute the contract automatically if conditions required have been met. Smart contracts are generally the most vulnerable points for cyber attack and technology failures. Like any other software code, smart contracts require robust testing and adequate controls to mitigate potential risks to blockchain-based business processes. The smart contracts that are defined on a blockchain network will apply in a consistent manner to all participants across the network. Therefore, these smart contracts will have to be capable of exception handling, and the consequences of these exceptions in the form of a programmatic output on the blockchain framework will have to be tested thoroughly within the network for adherence to business procedure arrangements.

Smart contracts may be sensitive and prone to security breaches and improper administration. Participant entities or the network administrator will need a strong change control process to deploy new or update already deployed smart contracts. Oracles are entities that exist outside the blockchain framework but feed data to the network, which could trigger the execution of the smart contracts within the network. The biggest risk to a blockchain framework may lie within these oracles as these could be subject to malicious attacks to corrupt the data being fed to the blockchain. This could cause very important risk domino effect across the entire network.

<u>Under optimized smart contracts</u>

When a user interacts with a smart contract deployed for instance in Ethereum, a certain amount of gas as transaction is charged. Unfortunately, some smart contracts' development and deployment are not adequately optimized. In the literature, there are studies which identify gas-costly patterns and group them

into 2 categories: useless-code related patterns, and loop-related patterns. They propose a tool named Gasper [8], which can automatically discover 3 gas-costly patterns in smart contracts: dead code, opaque predicate, and expensive operations in a loop. Leveraging Gasper, they find that significant percentage of smart contracts deployed in Ethereum have at least one of these 3 patterns. The details are as follows:

(1) Dead code: when some operations in a smart contract will never be executed, but they will still be deployed into the blockchain. Since in the smart contract deployment process the consumption of gas is related to byte code size, the dead code will cause additional gas consumption.

(2) Opaque predicate. For some statements in a smart contract, their execution results are always the same and will not affect other statements and the smart contract. The presence of the opaque predicate causes the EVM to execute useless operations, thereby consuming additional gas.

(3) Expensive operations in a loop. It refers to some expensive operations within a loop, which can be moved outside the loop to save gas consumption.

Transaction Privacy Leakage:

Since the users' behaviors in the blockchain are traceable, the blockchain systems take measures to protect the transaction privacy of users. In blockchains such as Bitcoin and Zcash, they use one-time accounts to store the received crypto currency. Moreover, the user needs to assign a private key to each transaction. In this way, the attacker cannot infer whether the crypto currency in different transactions is received by the same user.

Data confidentiality and Key management risks:

While the consensus protocol immutably seals a blockchain ledger and no corruption of past transactions is possible, it's still susceptible to private keys theft and the takeover of assets associated with public addresses. Digital assets could become irretrievable in the case of accidental loss or private key theft, especially given the lack of a single controller or a potential escalation point within the framework. The consensus protocol requires that all participants in the framework can view transactions appended to the ledger. While the transactions in a permissioned network could be stored in a hashed format so as to not reveal the contents, certain metadata will always be available to network participants. Monitoring the metadata can reveal information on the type of activity and the volume associated with the activity of any public address on the blockchain framework to any participant node.

Upon conducting this exercise it is time to properly provide a detailed list of threats that may act at the Application level; therefore, a depiction of the interfaces that will be involved in the related information exchanges (see Figure 7) helps to understand what we deal with.

[8] https://ieeexplore.ieee.org/document/7884650

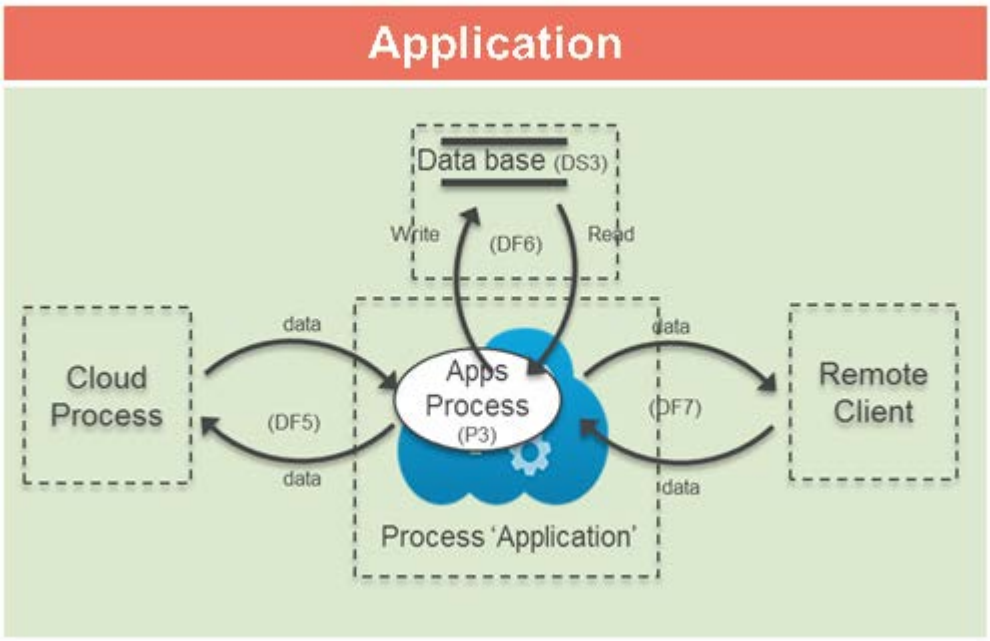

Figure 7. Interfaces for Applications

As it is the case with previous topics, this reference helps to better understand the exercise reflected in Table 12 which presents the relevant threats affecting the Application layer of the M-Sec framework, along with their associated risk rating.

Table 12. M-Sec Application related threats

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RATING
Thr.App.1	Libraries and modules on which the application is reliant, can be compromised or replaced by malicious versions. (they can be affected by the same threats as the application itself)	S, D, T	Lower Levels, DS4, communication links, Digital assets, Application Logic		All Use Cases	1	3	3
Thr.App.2	Other malicious agents can issue requests and data on behalf of the application.	S (e.g. IP Spoofing)	DF7, DF9, DS4, Digital Assets, Application Logic	Connected Care	Use Case 2	3	5	15
Thr.App.3	Malicious agents may have read access to the data the application is processing, and results.	S, I	DF7, DF8, DS4, Digital Assets, Application Logic	Park Guide, Connected Care, Smile City Report	Use Case 1, Use Case 2	3	5	15
Thr.App.4	Malicious agents may have write access to the data the application is processing. Being able to change it and produce unpredicted states	T	DF7, DF8, DF9, DS4, Digital Assets, Application Logic	Connected Care	Use Case 2	3	5	15
Thr.App.5	Data sources may be replaced, feeding erroneous or malicious data into the system workflow. E.g: Buffer overflow; cross-site scripting; SQL injection; canonicalization	E, T	DF7, DF9, DS4, Digital Assets	Connected Care, Smile City Report	Use Case 2	1	3	3

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RATING
Thr.App.6	Compiled, binaries or bytecode of the application may be corrupted or maliciously altered for execution.	T	DS4, Digital Assets		All Use Cases	3	5	15
Thr.App.7	Legit requests may have undesirable effects.	T, D	Application Logic		All Use Cases	3	5	15
Thr.App.8	The user may be convinced to perform actions that expose their data, or the application workflow (Social Engineering)	R	DS4, Application Logic, DF7, DF9		All Use Cases	1	3	3
Thr.App.9	Stored Data may be compromised. Either the cryptographic keys are not secure enough; the algorithms, the storage container is compromised or there might be some issue in the whole workflow.	T	DS4, Digital Assets,	Park Guide, Connected Care, Smile City Report	Use Case 1, Use Case 2	1	5	5
Thr.App.10	The user account is compromised. Either because the user has released, forgot, or shared her/his credentials, or because the account is meant to be shared amongst several users.	S	DF9, DF8, DS4, Digital Assets, Application Logic	Park Guide, Smile City Report	All Use Cases	3	5	15
Thr.App.11	The application may be compromised, because there is some extreme cases that are not considered, or certain assumptions make it susceptible to get to unstable states	I	Application Logic, Digital Assets	MTSA	All Use Cases	1	5	5
Thr.App.12	The application (or platform) does not provide log of the transactions and/or execution trace. Leaving potential attacks un accounted.	R	DS4, Digital Assets, Application Logic	Park Guide, Connected Care, Smile City Report	Use Case 1, Use Case 2	1	5	5
Thr.App.13	The application uses un registered communications (not known to the	E	DF9, DF7, DF8, DS4,		All Use Cases	1	5	5

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RATING
	underlying platform) or without relation to the functioning of the app itself.		Digital Assets					
Thr.App.14	The application does not use the appropriate authorization mechanisms, or these mechanisms can be easily circumvented	S	DF7, DF9, Application Logic, Digital Assets	Connected Care	Use Case 2	1	5	5
Thr.App.15	The application does not use the appropriate authentication mechanisms, or these mechanisms can be compromised (e.g.: key logger, un secured password storage or transmission, etc.)	S	DF7, DF9, DS4, Application Logic, Digital Assets		All Use Cases	1	5	5
Thr.App.16	Vulnerabilities-flaws in smart contracts	T,R,I,D	Application	Blockchain app / Smart contract	All Use Cases	3	5	15
Thr.App.17	Under-optimized smart contracts	T,R,I,D	Application	Blockchain app / Smart contract	All Use Cases	3	3	9
Thr.App.18	Transaction privacy leakage	T,R,I,D	Application	Blockchain app / Smart contract	All Use Cases	3	3	9
Thr.App.19	Misunderstanding of the agreement of applications.	I	Application	All apps	All Use Cases	1	3	3
Thr.App.20	Personal information and facial images are mistakenly uploaded in the marketplace and traded.	I	Application	Marketplace	Use case 5	3	5	15
Thr.App.21	Malicious agents may make fake transactions.	I	Application	Marketplace	Use case 5	3	5	15

THREAT #	DESCRIPTION	STRIDE THREAT CLASS	INTERFACES	M-SEC ASSET	SOURCE	PROBABILITY	CRITICALITY	RATING
Thr.App.22	Malicious agents may upload face data.	I	Application	Marketplace	Use case 5	3	5	15

End-to-end Risks

An IoT system consists of difference devices ranging from IoT devices at the environment side, edge processing servers, cloud servers, and application software at the users' side. The data are generated at the IoT devices, processed in the network, and delivered to the application software. The risks focused here relate to this data flow in that the data and services must be secure all through the end-to-end interactions. Since the system consists of different subsystems, multiple security issues exist in those subsystems. The end-to-end risks are in one sense the sum of all the subsystems risks, but more importantly, those risks must be seen end-to-end basis instead of point-to-point basis. In the following, the risks are broken down into the major attributes of security, and discussed in depth.

Confidentiality

Unauthorized disclosure of information should not happen. End-to-end confidentiality means that the data are encrypted at the IoT devices side, all the way through the edge and cloud servers down to the application. This divulge the following risks in an IoT system.

（1） First, if the data are decrypted in a subsystem between the IoT devices and applications, such as inside a cloud server, this becomes a risk. Since the subsystem may be attacked, the data should be kept encrypted.

（2） Second, if a network is unencrypted, this becomes a risk. Even if the data are encrypted, they are eavesdroped, and may be decrypted by malicious attackers.

（3） Finally, at the applications side, the device where the app runs may be attacked. Therefore the end-user devices should be protected properly.

Integrity

Improper system alterations should not happen. Even if an IoT system seems running properly, a subsystem may be altered maliciously. Credentials stored in the subsystems may be altered. Cryptographic keys may be altered, if they are written outside the software, e.g. in a file.

Availability

Correct services should be ready all time. Any form of incorrectness is considered to be a risk in an IoT system. Such risks can come from

(1) development faults such as incorrect software/hardware design and implementation, insufficient software/hardware performance, and so forth. Even if the software/hardware is successfully built, they should be updated in runtime to cope with additional requirements.
(2) physical faults such as destruction of a hardware such as IoT devices being physically attached to make them malfunction.
(3) Interaction faults such as attacks to cloud servers to steal data, tamper data, or stop the service.

All of these faults may exist in different subsystems in an IoT system and just one of them can harm the IoT system functioning property.

Privacy, GDPR, PIPA, Ethics related Risks

Nowadays, the widespread use of Internet and social networks, the advanced state of data analysis and the advent of the Internet of Things, have completely changed the way users and companies exchange information.

As a result, the new EU Data Protection Regulation seeks to lay down the foundations of a privacy law that suits the modern technology. It must be kept in mind that the GDPR will apply not only to European companies, professionals, data processing controllers and entities catering to the European community, but to all non-EU businesses and professionals who process data as part of any services aimed at European citizens.

The GDPR focused on an analysis of the impact of privacy, the right to information, consent, transparency and security, as well as to guarantee citizen's rights, regarding privacy and data protection.

The impact of privacy

This aspect is analysed by a process known as Privacy by desing, that aims to look at the impact of data processing carried out by companies, proffesionals and enterpreneurs with the goal of taking appropriate preventive measures to protect the user against any misuse of their personal data, thus preventing companies infringing upon their privacy.

Right to information

Until now, it has been a requirement to provide information about who precisely is behind each instance of data processing as well as how that information is being used. However, with the new GDPR, entities will have to ensure that the information is far more detailed than it has ever been. In the handling of personal data belonging to natural and legal persons (among others), the following information must be submitted the identity of the controller and of his or her representative, if any; the purposes of the processing for which the data are intended and legal obligation to which the controller is subject; the purposes for which the personal data will be used; and the period during which the personal data may be kept and, where this is not known, the criteria used by the controller to determine the said period. It is also important to note that the legal notifications, relevant contract clauses and a website's privacy policy should be as straightforward as possible, written in a clear, accessible way for the benefit of all users.

Consent

The new GDPR renders it crucial for interested parties to produce an explicit declaration or affirmative action that demonstrates conformity to sharing personal data, ensuring that any consent is indeed "unambiguous". In this sense, online companies and platforms will need to be endowed with systems that easily allow them to prove that explicit consent has been given. Implicit affirmative consent will not suffice: any gesture of consent must be tangible.

Transparency

As previously commented, any instance of data processing should come with easily intelligible terms and conditions. In order to be as transparent as possible, users should be able to exercise their full rights at all times, not only through being able to access, modify, erase or contest any of their information, but also through exercising their right to complain, in line with the well known "right to be forgotten".

Security

Companies and organizations are required to proactively implement security measures that guarantee that the company's infrastructure disposes of the correct processing and storage methods for client and user data. The main measures that should be adopted are: establishing secure access to the company's system or database; establishing adequate backup procedures; and taking measures to avoid data leaks, the installation of malware and any other associated risks occurring, such as attacks by crackers, denial-of-service attacks or system failures, etc. Companies are always required to report any type of incidents to the authorities and affected users when their privacy or personal data is at risk.

3. Privacy Enhancing Technologies

New information technologies change the privacy and data protection risks we are facing in a bilateral way: although new risks (e.g. through ease of search, cheap data storage) emerge, technology can also help to minimise or avoid risks to privacy and data protection.

The idea of shaping technology according to privacy principles has been discussed since many years, addressing among other the principles of data minimisation, anonymisation and pseudonymisation. This led to the term Privacy Enhancing Technologies (PETs), which covers the broader range of technologies that are designed for supporting privacy and data protection.

3.1 Existing Private Enhancing Technologies

There is no universally agreed taxonomy for PETs. For example, the European Union Agency for Network and Information Security (ENISA), in its work on a PETs controls matrix, identifies four major categories of technology: secure messaging, virtual private networks, anonymizing networks and anti-tracking tools for online browsing. Other researchers have characterized PETs *"according to their technical contributions"* (e.g., anonymous communication, and privacy preserving data mining).

For the purposes of this deliverable, the taxonomy for privacy-enhancing technologies described below is a way of classifying these technologies based on the functionality/capabilities that they provide to an end user. This particular taxonomy has been chosen because it provides a fairly granular way of categorizing the various tools and techniques that have been identified during our review, using terms that often appear in common usage or in the media. It also helps identify areas where additional research and development is required. The principal drawback is that some tools and techniques provide more than one capability, making it somewhat difficult to neatly categorize them.

PETs are intended to allow users to protect their (informational) privacy by allowing them to decide, amongst other things, what information they are willing to share with third parties such as online service providers, under what circumstances that information will be shared, and what the third parties can use that information for. They do this by providing one or more of the following functions/capabilities.

3.1.1 Informed Consent

When an individual discloses his or her personal information to commercial and other entities, he or she also grants, sometimes explicitly, sometimes implicitly, **consent** for it to be used for one or more purposes. Consent is a key principle of most data protection/privacy legislation. Although the specific language varies, a key element of consent is that it be informed (i.e., based on a clear understanding of what the individual is consenting to). Subsequent control over the storage, use, and onward sharing of that information relies on the notion of trust that the given consent will be respected. Unfortunately, the reality is that, given the complexity of the policy language, the complexity of the business ecosystem behind the organization with whom the individual is dealing, and similar factors, this trust is sometimes misplaced.

One way for this trust to be restored is through the use of a technique known as *"data tagging"*. In data tagging, a user's personal information is labelled or tagged with instructions or preferences specifying how the data should be treated by service providers. These preferences can be expressed in a machine readable format using a privacy policy language, and automatic mechanisms have been proposed to ensure that service providers follow the instructions.

Sticky policies are an example of data tagging. Sticky policies technically enforce preferences when personal data is shared across multiple parties. One way to enforce this is through the use of encryption. The EnCoRe (Ensuring Consent and Revocation) project proposed an architecture[9] where encrypted personal data, with a machine-readable policy stuck on, can only be decrypted and read by entities that abide by the policy rules. A trust authority enforces this by verifying compliance and only distributing decryption keys to those services that adhere to the policies.

Sticky policies are an integral part of certain privacy policy language proposals such as PPL (PrimeLife Policy Language)[10] and E-P3P (Platform for Enterprise Privacy Practices)[11]. PPL is based on XACML (eXtensible Access Control Markup Language)[12] [REF] and is used to grant service providers access to data as long as the organization's policy is compatible with the user's privacy preferences. E-P3P is a privacy-specific access control language that allows organizations to design and deploy machine-readable privacy policies, including identifying opt-in or opt-out choices (depending on the nature of the information) and placing restrictions on access to personal information, and design access control policies to give effect to the privacy policies.

Data tagging and sticky policy research has been ongoing since 2002, but the work remains at the proof of concept stage with few commercial deployments. In general, machine-readable, automated policy languages have had very limited success, perhaps due to complexity, a lack of interoperability and little demand for their capabilities. Most recently, Microsoft has discontinued all support for P3P in their Windows 10 browsers.

3.1.2 Technical Enforcement

In those instances where individuals are able to negotiate the terms and conditions of a service, PETs in this category provide individuals with the possibility of having these terms and conditions **technically enforced** by the infrastructures of online service providers and merchants (i.e., not just having to rely on promises, but being confident that it is technically impossible for service providers to violate the agreed upon data handling conditions). Technical enforcement of negotiated terms and conditions can be accomplished in a number of different ways, many of which are currently in use, albeit for different purposes (this list is not intended to be exhaustive):

---

[9] EnCoRe project proposed architecture, https://www.hpl.hp.com/breweb/encoreproject/deliverables_material/D2.1%20EnCoRe%20Architecture%20V1.0.pdf

[10] The PrimeLife Policy Language (PPL) was developed by the PrimeLife Consortium (http://primelife.ercim.eu/) as part of the European Commission's 7th Framework Project.

[11] For more information on the Platform for Enterprise Privacy Practices (E-P3P), see, for example, "Platform for Enterprise Privacy Practices: Privacy-enabled Management of Customer Data" [http://www.semper.org/sirene/publ/KaSW1_02.EP3P4PET.pdf], Karjoth, G., Schunter, M., and Waidner, M., presented at the 2nd Symposium on Privacy Enhancing Technologies, San Francisco, 14 – 15 April 2002, accessed 2 October 2017.

[12] XACML, https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

a. network monitoring: passive or active monitoring of network activity to compare the activity against the agreed terms and conditions (e.g., Wireshark[13], Fiddler[14], and so on). Some tools provide real-time prevention of privacy leaks;

b. endpoint event detection: a category of tools and solutions that focus on detecting, investigating, and mitigating suspicious activities and issues on hosts and endpoints (e.g., McAfee Active Response, Symantec Endpoint Protection, and so on);

c. web transparency tools: these tools are primarily intended to provide a user with information about the intended collection, storage and/or data processing of their personal information, or to help the user determine the potential impact of data profiling. Such tools include ad blockers (e.g., Adblock Plus and Ghostery), and tracking blockers (e.g., Privacy Badger); and

d. enterprise digital rights management: access control technologies that try to control the use, modification, and distribution of copyrighted works (such as software and multimedia content), as well as systems within devices that enforce these policies (e.g., ContentGuard, Digimarc, and so on).

3.1.3 Remote Audit of Enforcement

PETs in this category provide individuals with the ability to remotely audit the enforcement of the terms and conditions offered by online service providers and merchants. While the term is most frequently applied to audits of an organization's financial information, other areas which can be audited include Governance, Compliance and Risk (GRC) and internal controls. An audit involves the gathering and analysis of information relevant to specified objectives, scope and criteria. While this information has traditionally been gathered in the form of onsite interviews, document reviews and through observation of processes or people, some of this information gathering can now be done remotely.

One way to facilitate the auditing of an organization is for that organization to pre-emptively publish information concerning their policies, procedures and practices. For example, timely, accurate statistical information from private sector firms on government requests for and access to personal information – in the form of clear transparency reports at regular intervals – can form the basis for rational consumer choices and build consumer confidence in a growing digital economy and its interface with the state for law enforcement and security purposes.

Another way for individuals to *"audit"* an organization is for the organization to undergo certification against a trust mark, defined as *"electronic labels or visual representations indicating that an e-merchant has demonstrated its conformity to standards regarding, e.g., security, privacy and business practice"*. Organizations that offer certification against a trust mark often make information about the trust mark, and the criteria an organization needs to satisfy to obtain the mark, available on their website. Individuals can then research the trust mark, as well as the trust mark provider, and decide if they are prepared to share their personal information with the website in question.

As useful as trust marks might be in helping establish trust in an organization, trust marks have their limitations. For example, a privacy trust mark (e.g., such as the ones issued by TRUSTe, now TrustArc[15]) does

[13] Wireshark network protocol analyzer, https://www.wireshark.org/
[14] Fiddler HTTP debugging proxy server, https://www.telerik.com/fiddler

not necessarily guarantee that the organization has implemented specific technical security standards or processes (such as basic traffic encryption or infrastructure vulnerability testing) as there may be more than one way to meet the requirements of the trust mark.

3.1.4 Use of Legal Rights

Many data protection/privacy laws provide individuals with certain rights, including the right to access the information about them that an organization holds, the right to challenge the accuracy and completeness of that information, and the right to have it amended as appropriate. Typically, exercising these rights requires individuals to send a written request to an organization and then wait for the organization to respond. One way to assist individuals in exercising their right is to automate the request process for them.

In 2014 the Citizen Lab, in partnership with Open Effect and Open Media, launched the original version of the Access My Info (AMI) tool[16]. AMI is a step-by-step wizard that results in the generation of a personalized formal letter requesting access to the information a provider stores and utilizes about a person. The original version only allowed users to generate a letter to telecommunications companies. An improved tool, re launched in June 2016, provides individuals with the ability to send formal requests to a broader range of organizations, including those that provide fitness trackers and dating applications.

3.2 Private Enhancing Technologies trends

Currently, in the big data world we are living in, PETs must exploit different computational and mathematical approaches with the purpose of extracting data value in order to unleash its full scientific and social potential, without jeopardizing the privacy and security of this information.

Advanced data analysis has great advantages for users, institutions and society as a whole, but at the same time it, opens up new scenarios in which privacy, anonymity and data security are put at risk. Different initiatives across the world are researching a series of cryptographic methods that allow data to be analyzed and shared without exposing their content to third parties.

Some of the most up to date were discussed in a recent report of the World Economic Forum, where the particular role of PETs in the financial sector was analyzed[17]. The PETs introduced in this context, which could be exported to other fields, are:

- Homomorphic Encryption

Homomorphic encryption is a form of encryption that allows certain computations on encrypted data, generating an encrypted result which, when decrypted, matches the result of the same operations performed on the data before encryption. It might be used in particular to securely outsource certain specific operations on sensitive data to the cloud, or to another third party organisation. It can also be used in combination with other PETs to safely share data.

---

[15] TrustArc, https://trustarc.com/
[16] AMI by Citizen Lab, https://accessmyinfo.org/
[17] World Economic Forum, "*The Next Generation of Data-Sharing in Financial Services: Using Privacy Enhancing Techniques to Unlock New Value*", September 2019, http://www3.weforum.org/docs/WEF_Next_Gen_Data_Sharing_Financial_Services.pdf

Homomorphic encryption can be used to analyse data in circumstances where all or a part of the computational environment is not trusted, and sensitive data should not be accessible. It is applicable where the computation required is known and relatively simple. Homomorphic encryption provides confidentiality and can be used to address the problems of *'insecurity'* and *'exposure'*, and the risk of revealing sensitive attributes related to individuals or organisations, in a dataset or output. Figure 7 depicts roughly its functioning.

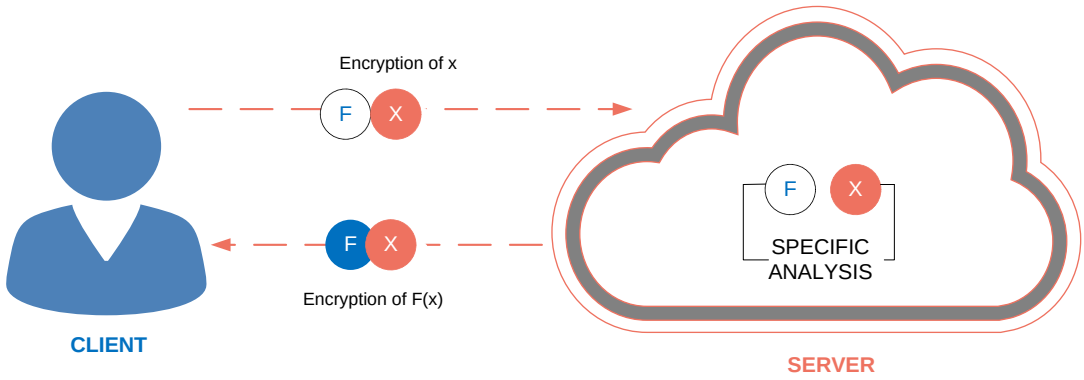

Figure 8. Homomorphic encryption depiction

As hinted, its practical applications are limited to the volume of data, since it can only be used to operate with limited amounts of information.

- Secure Multi-Party Computation

This cryptographic technology is actually a subcategory of the previous one, which allows complex computational or analytical operations to be performed on a larger volume of encrypted data; which in turn allows models of *"machine learning"* to be applied to them.

Its use is already widespread in companies such as Google and Facebook, and is present in products such as the machine learning Tensor Flow tool[18], which enables models to be trained with encrypted third-party data. For this purpose, companies share their encrypted data with a third party, who analyzes it and sends back the results of the analysis, without compromising the privacy of its content.

One of the fields with the most evident application for this is healthcare. Nowadays, there are already projects that improve diagnosis by using image analysis based on this technology, so that the systems can learn while keeping the patients' private data from being disclosed.

- Federated Learning

This method goes one step further than the rest, and it enables automated learning models to be trained from data that does not even have to leave the company or the device it was generated on. A very useful approach for Federated Learning appears in the fields of the Internet of Things and in the advanced analytics one.

This technology, which large companies such as Google are already researching, could also help, for example, to train the intelligence systems of virtual assistants by collecting data on site on the different

[18] TensorFlow, https://www.tensorflow.org/

devices connected to a collaborative learning network, but in a way that keeps this data from leaving the device on which it is generated. Therefore the only thing that is shared is the data generated that will be relevant for the model's training, without any personal or sensitive data, which helps the learning system grow but does not contain users' private information.

- Zero-Knowledge Proofs

This technology allows verifying whether there is validity to information, without exposing the data that proves it. This is possible thanks to a series of cryptographic algorithms through which a "tester" can mathematically prove to a "verifier" that a computational statement is correct, without revealing any data.

Its applications are numerous for the creation of opportunities in the banking and insurance sectors, in which it could facilitate access to products or services for which private customer information is required, while ensuring that they do not expose their data.

- Differential Privacy

This cryptographic system allows a "*random noise*" layer to be added to a data set, so that it is impossible to extract specific information about each individual piece of information. Thus, it is possible to share the results of applying an automated learning model to a data set with a third party, while keeping the analyzed data private.

- Trusted Execution Environments

A Trusted Execution Environment (TEE) is a secure area inside a main processor. Figure 4 shows TEEs are isolated from the rest of the system, so that the operating system or hypervisor cannot read the code in the TEE. However, TEEs can access memory outside. TEEs can also protect data 'at rest', when it is not being analysed, through encryption.

Like homomorphic encryption, TEEs might be used to securely outsource computations on sensitive data to the cloud. Instead of a cryptographic solution, TEEs offer a hardware based way to ensure data and code cannot be learnt by a server to which computation is outsourced. TEEs are a good place to store master cryptographic keys, for example. Figure 8 sketches how a TEE creates a secure area inside a processor.

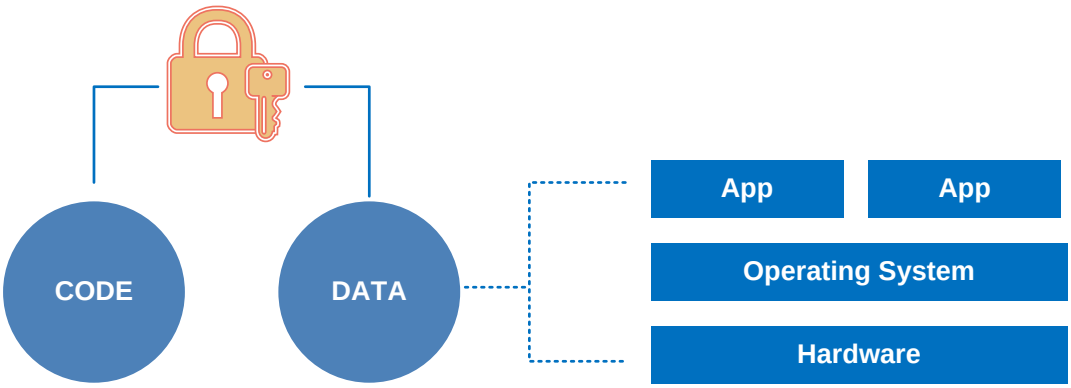

Figure 9. TEE depiction

In addition, TEEs can support any type of analysis. They have a low cost to utility: the actual computation is performed on the unencrypted data, and no noise needs to be added to the data.

3.3 M-Sec PETs of choice

Upon getting to know the variety of PETs at disposal, it is required to find out which ones are more well suited for a context such as the one posed by M-Sec.

It is worth noting that some PETs might be better suited for use by organisations (*Business-to-Business*; B2B), and others for use by individuals (*Business-to-Consumer*; B2C). For example, cloud providers may want to employ secure hardware or techniques based on encryption to protect code and data on their platform, whereas individuals may benefit from using personal data stores and other PETs designed for individuals.

Within organizations or in collaborative projects such like this, PETs could support cross-team working, by enabling different internal teams to work on data that they otherwise would not be able to match or view because of data protection rules. In addition and as part of a research and development context the safe sharing of data could also lead to innovations benefiting citizens. However, this would not detract from the need to assess whether it is legal or ethical to carry out an analysis or give access to the data in the first place.

Before taking any decision, it is worth noting that when using PETs, there are trade-offs. Privacy engineers say that PETs incur a cost in terms of 'utility'. In the context of different technologies, the cost in utility might be of a different nature. For example, with differential privacy adding noise to a dataset entails a loss of some useful information so there is a cost in terms of accuracy. In the case of PETs where computation happens on encrypted data, such as homomorphic encryption and secure multi-party computation, the main cost to utility is in terms of computation resources (time, computing power). In order to negotiate these trade-offs, the consortium needs to have a clear idea of what information or value they are trying to protect, and they need to determine the potential benefits and costs of different PETs so that M-Sec system can be optimised for this.

According to the scope of the project, the PET mainly applied will be the Informed Consent, establishing in every use case that requires so the process for getting permission before the user may join the experience and thus granting M-Sec is awarded the acquiesce to disclosing users' personal information. In addition, the use of TPM provides the ability to do crypto calculation, quite similar to the role a TEE plays; nevertheless, the TPM is physically isolated from the rest of the processing system.

4. Risks mitigation

Once the threat analysis process is completed, including how a series of potential threats may affect M-Sec architecture components and thus the use cases themselves, there is a chance to discuss the components, techniques and methods that will help M-Sec to proceed with the risk mitigation and achieve a substantial drop in the corresponding risk rating.

In an effort to keep the consistency, these mitigation activities link to every threat in the lists previously presented in Section 2.

IoT Risks Mitigation

M-Sec platform will integrate components with the mission of strengthening the IoT layer, which will become one of the security layers in the overall platform, providing the needed security and reliability for IoT as follows:

- **IoT devices** with increased security, an asset that further strengthens the current state of the art Security provision in IoT devices on a hardware level.
- **IoT platforms share a comm**on ground, but also exhibit considerable dissimilarities in terms of architecture and concerning how the services can be implemented. However, a common security criterion can be useful to apply in they all and specifically in M-Sec. The security criteria can be based on the five following topics:
    - **Authentication**: The authentication process provides a way of identifying an entity within the system. In the context of IoT, the produced data are massive, causing several security and privacy issues, especially regarding the authentication among the devices, the users and the system itself. The following sub-criteria can be considered:
        - Authentication protocol between IoT devices and IoT platform.
        - Authentication protocol between the IoT platform and users.
        - Authentication protocol among the components of the IoT platform.
    - **Encrypted information management**: The encryption process takes care of encoding a message or information so that only authorized entities can access it. Due to the amount of information and the resource-constrained nature of the IoT devices, the encryption assumes a fundamental role. In this case, we analyzed the encryption at two levels:
        - Encryption for data at rest (i.e., data physically stored and inactive)
        - Encryption for data in transit (i.e., data transmitted among the entities)
    - **Authorization**: Once a user has been successfully authenticated in the system, the authorization process determines whether the user has the rights to access a given resource or to execute an activity. As in the case of authentication, authorization strategies represent a strong requisite in the IoT ecosystem, since unauthorized intruders can perform malevolent actions, such as compromising the integrity of the system by maliciously modifying its data. This feature can be detailed in two sub-criteria, as follows:
        - User authorization to perform operations within the IoT platform.

- Authorization of the IoT devices to perform operations among the components of the IoT platform.

- **Accounting**: It measures the amount of resources a specific user consumes during their access. Examples of resources include the session time or the data which the user has sent and/or received during a session. Given the huge number of performed operations within the IoT platform, this parameter is also crucial from a security perspective. The following sub-criteria may be followed:
  - Accounting for operations that users perform over the data and IoT devices (e.g., read, write, aggregation, etc.)
  - Accounting for operations created for IoT services components on data and IoT devices (e.g., accounting, disassociation, update, etc.)
- **Anomaly detection:** It refers to the ability of the system to spot anomalies among the normal activities, which may indicate the presence of a security incident. In this context, this criterion is related to the IoT platform capability to detect anomalies in IoT service's state or in the normal operation of its components. We consider this characteristic of primary importance in a full-fledged IoT framework, so that the security incidents can be reported in a timely fashion, and possible countermeasures can be undertaken.

Table 13 reminds the threats related to this topic, as well as the threaten M-Sec assets and its original risk rating, and offers some clues on which kind of mitigation actions will be taken.

Table 13. M-Sec mitigation for IoT devices related threats

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.IoT.1	Data stored in the device can be read by an intruder	EnMon, Crow, Caburn	9	TPM will be designed to reduce the probability by securing the IoT device itself.
Thr.IoT.2	An unauthorized party can modify data on the device	EnMon, Crow, Caburn	9	TPM will encrypt data to reduce this risk.
Thr.IoT.3	Man in the middle attack: a third party puts itself between the entity that communicates with the device and the device itself, without them noticing (so that both actually communicate with the intruder)	EnMon, Crow, Caburn	9	Only PCB with hard coded sensors. Solution will come via HW and SW. TPM will encrypt data to reduce this risk. Serial connection is to be clamped and Locked. Physical security to mitigate risk by lowering likelihood

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.IoT.4	Unauthorized modification of configuration parameters of the device or the sensor	KEIO Mobile Sensing Platform	9	Hard coded circuitry. Someone needs to steal and replace with modified PCB. Few affected IoT devices and/or sensor boxes, not critical. Security cameras in the surroundings lower probability
Thr.IoT.5	An attacker can overload the device by injecting many requests	EnMon, Crow, Caburn	-	No services active, so no requests can be processed
Thr.IoT.6	Jamming of the wireless communication link	KEIO Mobile Sensing Platform	-	Not applicable as there is no Wireless interface
Thr.IoT.7	Accidental or intentional physical damage to any device part may cause device failure	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	9	Risk will be mitigated by physically putting devices in secure enough locations and by Physically clamping and locking device securely. Few sensor failure is not critical.
Thr.IoT.8	Insecure firmware update mechanism: the firmware has been retrieved at a non-valid source	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	-	Not applicable as there is no firmware
Thr.IoT.9	If installed, malware has full access to the whole device	EnMon, Crow, Caburn	-	No OS Not applicable as there is no OS
Thr.IoT.10	If installed, malware has access to data in the device	KEIO Mobile Sensing Platform	-	Not applicable as there is no OS
Thr.IoT.11	If installed, the malicious firmware may cause device operation failure	EnMon, Crow, KEIO Mobile Sensing Platform	-	Not applicable as there is no OS
Thr.IoT.12	Insecure firmware update mechanism: the firmware is corrupted	EnMon, Crow, KEIO Mobile Sensing Platform	-	Not applicable as there is no firmware

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.IoT.13	A device designed to be used by several users and keeping history per user discloses information on the other users (spontaneously or after wrong manipulation).	EnMon, Crow, KEIO Mobile Sensing Platform	-	Not applicable as there is no such capability
Thr.IoT.14	A weak authentication method is very likely to be used (short and simple passwords, if any), opening a door to data and device exposure (breaches to integrity and confidentiality).	EnMon, Crow, KEIO Mobile Sensing Platform	-	Not applicable as there is no OS and firmware. Where applicable, strong complex passwords will be set.
Thr.IoT.15	The device was reset to its default settings, which does not include security.	EnMon, Crow, KEIO Mobile Sensing Platform	-	Not applicable as there is no OS and firmware
Thr.IoT.16	Nobody is responsible for device maintenance.	EnMon, Crow, KEIO Mobile Sensing Platform	3	M-Sec partners will play this role and assign a responsible person. Partners have already assigned their responsible for the maintenance to lower the likelihood and impact.
Thr.IoT.17	Nobody is responsible for system management and maintenance (e.g. system: device network)	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	3	M-Sec partners will play this role and assign a responsible person. Partners have already assigned their responsible for the maintenance to lower the likelihood and impact.
Thr.IoT.18	Attack on Power Management	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn	9	No Firmware or data storage. Physical security & clamping to mitigate risk by lowering likelihood.
Thr.IoT.19	A visitor is playing with the device and wrongly records measurements that are not those of the intended user.	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	15	Certain monitors can have wrong measurements if someone touches it, even without knowing it.
Thr.IoT.20	Old persons may not know how to handle electronic devices efficiently. They make errors and quit easily.	EnMon, Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	15	One of the conclusions of the survey conducted at the beginning of the project among users of tele assistance service, and which had been anticipated by municipal social services officers as well as tele assistance company staff, was users' concern about the difficulty in the use of technological devices (more than 57% were worried about this difficulty). Furthermore, more than

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
				85% of them are over 71 years old. Therefore, devices that do not require human interaction, such as door/window opening sensor, smart plug sensor, were selected.
Thr.IoT.21	An old person denies having recorded a measurement (e.g. blood pressure rate).	Crow, KEIO Mobile Sensing Platform, Caburn IoT Devices	9	In order to lower the risk, devices to be deployed do not require human interaction.
Thr.IoT.22	A member of caring personnel denies having administered a treatment.	Caburn IoT Devices	-	Out of the scope of Use Case 2

Communication Level Risks Mitigation

Table 15 reminds the threats related to this topic, as well as the threaten M-Sec assets and its original risk rating, and offers some clues on which kind of mitigation actions will be taken.

Table 14. M-Sec mitigation for Communication related threats

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENT / MITIGATION
Thr.Com.1	Communications may be eavesdropped	IoT Gateway, Caburn	15	Mitigate with Encryption, Authentication & physical security
Thr.Com.2	Unrestricted access to alter device configuration	IoT Gateway, Caburn	15	Authentication & physical security
Thr.Com.3	Data storage is readable without authentication	IoT Gateway, EnMon, Crow, Caburn	9	Authentication & Encryption
Thr.Com.4	Data storage is writeable without authentication	IoT Gateway, Caburn	9	Authentication & Encryption
Thr.Com.5	IoT physical interfaces (USB dongles, etc.) are removed	IoT Gateway	25	Physical security & interfaces to be clamped and locked
Thr.Com.6	Device is removed or put out of range	IoT Gateway, EnMon, Crow, Cabrun	15	Physical security & device to be clamped and locked. Few affected devices not critical.
Thr.Com.7	Gateway is unplugged to free a power plug	IoT Gateway, Caburn	25	Cables to be clamped and locked
Thr.Com.8	Gateway is stolen for reverse engineering	IoT Gateway, EnMon, Crow, Caburn	9	Physical security & Authentication
Thr.Com.9	Disclosure of encryption parameters for the communication channels	IoT Gateway, Caburn	15	Security Manager
Thr.Com.10	Attack on the security implementation	IoT Gateway	25	Implement IoT gateway security

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENT / MITIGATION
Thr.Com.11	Attack by replay or Man-in-Middle Attack	IoT Gateway, EnMon, Crow, Caburn	15	Encryption
Thr.Com.12	DHCP network parameters are compromised to spoof other processes (ARP, DNS)	IoT Gateway	3	3G ISP security exists
Thr.Com.13	Wireless signals are jammed (GPRS, WiFi, other)	IoT Gateway, Crow, Caburn	3	Disable WiFi in Field. Unlikely to jam 3G of all 60+ trucks on different routes during pilot
Thr.Com.14	An attacker gains access to the boot process using a physical port (serial console, JTAG)	IoT Gateway	9	Physical security & Authentication
Thr.Com.15	An attacker gains access to the operating system using a service on the system (telnet, FTP, SSH, etc.)	IoT Gateway, Caburn	15	Implement IoT gateway security
Thr.Com.16	A weak authentication method (short/simple/default passwords, if any), opening a door to data and device exposure (breaches to integrity and confidentiality).	IoT Gateway, Caburn	15	Strong Authentication - complex password setting (10-digits or more). Keio > there are many devices. It is impossible to reset PW frequently.
Thr.Com.17	If installed, malware has full access to the device or data	IoT Gateway, Caburn	15	IoT gateway security.
Thr.Com.18	Accidental or intentional physical damage to any device part may cause device failure	IoT Gateway, EnMon, Crow, Caburn	9	Risk will be mitigated by Physically clamping and locking device securely. Few sensors failure is not critical.
Thr.Com.19	The device was reset to its default settings, which does not include security.	IoT Gateway, Caburn	25	Enable security by default on reset
Thr.Com.20	Nobody is responsible for device management and maintenance.	IoT Gateway, EnMon, Crow	3	M-Sec partners will play this role and assign a responsible person. Partners have already assigned their responsible for the maintenance to lower the likelihood and impact.
Thr.Com.21	51% attack over blockchain	Quorum Blockchain	5	Avoid PoW blockchain and use instead permissioned - delegated consensus mechanisms
Thr.Com.22	Private key security - Public key encryption scheme	Quorum Blockchain	3	Permissoined blockchains can mitigate the specific threat more easily
Thr.Com.23	Data storage is physically removable	N/A	-	No removable storage exists

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENT / MITIGATION
Thr.Com.24	Removable data storage (SD card with OS) is stolen to extract security credentials	N/A	-	No removable storage exists
Thr.Com.25	Over consumption of resources (processor, memory) due to an unmonitored activity	N/A	-	Out-of-scope for Pilot 3.1
Thr.Com.26	Gateway OS is updated with a malicious update	N/A	-	Out-of-scope for Pilot 3.1
Thr.Com.27	If installed, the malicious firmware may cause device operation failure	N/A	-	Out-of-scope for Pilot 3.1
Thr.Com.28	The boot process is altered to run a rogue OS	N/A	-	Out-of-scope for Pilot 3.1
Thr.Com.29	The system is un-updated, leaking access to potential attackers	N/A	-	Out-of-scope for Pilot 3.1
Thr.Com.30	Pairing the gateway with non-legitimate devices	N/A	-	No pairing interface exists

Cloud Level Risks Mitigation

Table 14 reminds the threats related to this topic, as well as the threaten M-Sec assets and its original risk rating, and offers some clues on whick kind of mitigation actions will be taken.

Table 15. M-Sec mitigation for Cloud related threats

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.CD.1	Impersonation: A third party uses a false ID to gain access to the cloud	SoxFire	9	Strong Authentication
Thr.CD.2	An attacker may install a malware to access the data and whole cloud system	SoxFire	3	Protected in Keio's network
Thr.CD.3	Accidental or intentional physical damage to any cloud part may cause cloud service failure	SoxFire	3	Backup server exists
Thr.CD.4	Disruption of a global service (e.g. attack on power management)	SoxFire	3	Backup power exists
Thr.CD.5	Data (raw & processed, personal data) stored in the cloud can be read by an intruder	SoxFire	15	Encryption
Thr.CD.6	An unauthorized party gets access to device configuration information	SoxFire	3	Protected in Keio's network
Thr.CD.7	Attacker denies legitimate users access to infrastructure services	N/A	-	No Infra services running

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.CD.8	Attacker can poison cloud database and/or alters outgoing information	SoxFire	15	Encryption
Thr.CD.9	An unauthorized party can modify data stored in the cloud data base	SoxFire	5	Protected in Keio's network
Thr.CD.10	Disclosure of private services, access policies, cryptographic material, user information	SoxFire	5	Protected in Keio's network
Thr.CD.11	Attacker gains knowledge of sensitive exchange data	SoxFire	15	Encryption
Thr.CD.12	Alteration of the invocation of a service	SoxFire	3	Protected in Keio's network
Thr.CD.13	Attacker can disrupt communications	SoxFire	5	Protected in Keio's network
Thr.CD.14	Man in the middle attack: a third party puts itself between the entity (e.g. gateway or application) that communicates with the cloud and the cloud itself, without them noticing (so that both actually communicate with the intruder)	SoxFire	15	Encryption
Thr.CD.15	Wrong authorization information propagating from one server to another	N/A		Only one server
Thr.CD.16	User is involved in transactions/actions with a malicious peer	N/A		No peer exists
Thr.CD.17	A service critical for user's safety is disabled	N/A		No such service
Thr.CD.18	Nobody is responsible for cloud maintenance	SoxFire	9	KEIO is responsible for its servers in cloud.
Thr.CD.19	Nobody is responsible for system management and maintenance (e.g. cloud infrastructure)	SoxFire	9	KEIO is responsible for its servers in cloud.
Thr.CD.20	An old person denies having recorded a measurement (e.g. blood pressure rate)	N/A		No user dependent feature
Thr.CD.21	A member of caring personnel denies having administered a treatment	N/A		No user dependent feature
Thr.CD.22	Attacker changes the association between the device id / personal id and the corresponding device or an old person	N/A		Not Applicable
Thr.CD.23	Attacker changes configurations and prevents proper communication to an actuator	SoxFire	9	Protected in Keio's network

Application Level Risks Mitigation

Following the same approach as in Section 2, certain general mitigation measures at application level are listed in Table 16, followed by a series of good practices to avoid a serious impact .

Table 16. Generic mitigation measures at Application level

Treads (STRIDE)	Countermeasures (Mitigates)	Technology Examples
Spoofing	Authentication	Passwords, Digital signatures
Information Disclosure	Permissions, encryption	Passwords, PGP, SSL
Elevation of Privilege	Authorization	Sandboxes, Firewalls
Tampering	Permissions	Digital signatures, ACLs
Repudiation	Signatures, Logging	Digital signatures, Logging
Denial of Service	Fraud prevention	Firewalls, Load balancers

Injection

- Verify that the runtime environment is not susceptible to buffer overflows, or that security controls prevent buffer overflows.
- Verify that server side input validation failures result in request rejection and are logged.
- Verify that input validation routines are enforced on the server side.
- Verify that a single input validation control is used by the application for each type of data that is accepted.
- Verify that all SQL queries, HQL, OSQL, NOSQL and stored procedures, calling of stored procedures are protected by the use of prepared statements or query parameterization, and thus not susceptible to SQL injection
- Verify that the application is not susceptible to LDAP Injection, or that security controls prevent LDAP Injection.
- Verify that the application is not susceptible to OS Command Injection, or that security controls prevent OS Command Injection.
- Verify that the application is not susceptible to Remote File Inclusion (RFI) or Local File Inclusion (LFI) when content is used that is a path to a file.
- Verify that the application is not susceptible to common XML attacks, such as XPath query tampering, XML External Entity attacks, and XML injection attacks.
- Ensure that all string variables placed into HTML or other web client code is either properly contextually encoded manually, or utilize templates that automatically encode contextually to ensure the application is not susceptible to reflected, stored and DOM Cross-Site Scripting (XSS) attacks.

- If the application framework allows automatic mass parameter assignment (also called automatic variable binding) from the inbound request to a model, verify that security sensitive fields such as "*accountBalance*", "role" or "password" are protected from malicious automatic binding.
- Verify that the application has defences against HTTP parameter pollution attacks, particularly if the application framework makes no distinction about the source of request parameters (GET, POST, cookies, headers, environment, etc.)
- Verify that client side validation is used as a second line of defence, in addition to server side validation.
- Verify that all input data is validated, not only HTML form fields but all sources of input such as REST calls, query parameters, HTTP headers, cookies, batch files, RSS feeds, etc.; using positive validation (whitelisting), then lesser forms of validation such as grey-listing (eliminating known bad strings), or rejecting bad inputs (blacklisting).
- Verify that structured data is strongly typed and validated against a defined schema including allowed characters, length and pattern (e.g. credit card numbers or telephone, or validating that two related fields are reasonable, such as validating suburbs and zip or post codes match).
- Verify that unstructured data is sanitized to enforce generic safety measures such as allowed characters and length, and characters potentially harmful in given context should be escaped (e.g. natural names with Unicode or apostrophes, such as ねこ or O'Hara).
- Make sure untrusted HTML from WYSIWYG editors or similar are properly sanitized with an HTML sanitizer and handle it appropriately according to the input validation task and encoding task.
- For auto-escaping template technology, if UI escaping is disabled, ensure that HTML sanitization is enabled instead.
- Verify that data transferred from one DOM context to another, uses safe JavaScript methods, such as using *.inner Text* and *.val*.
- Verify when parsing JSON in browsers, *that JSON.parse* is used to parse JSON on the client. Do not use *eval()* to parse JSON on the client.
- Verify that authenticated data is cleared from client storage, such as the browser DOM, after the session is terminated.
- If at all possible, use library calls rather than external processes to recreate the desired functionality.
- Run time: Run time policy enforcement may be used in a whitelist fashion to prevent use of any non-sanctioned commands.
- Assign permissions to the software system that prevents the user from accessing/opening privileged files.
- For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side.
- When exchanging data between components, ensure that both components are using the same character encoding. Ensure that the proper encoding is applied at each interface. Explicitly set the encoding you are using whenever the protocol allows you to do so.
- When your application combines data from multiple sources, perform the validation after the sources have been combined. The individual data elements may pass the validation step but violate the intended restrictions after they have been combined.

Broken Authentication

- Use an authentication framework or library such as the OWASP ESAPI Authentication feature.
- Use multiple independent authentication schemes, which ensures that -- if one of the methods is compromised -- the system itself is still likely safe from compromise.
- Use industry standards to protect the credentials (e.g. LDAP, keystore, etc.).
- Verify all pages and resources by default require authentication except those specifically intended to be public (Principle of complete mediation).
- Verify that forms containing credentials are not filled in by the application. Pre-filling by the application implies that credentials are stored in plaintext or a reversible format, which is explicitly prohibited.
- Verify all authentication controls are enforced on the server side.
- Verify all authentication controls fail securely to ensure attackers cannot log in.
- Verify password entry fields allow, or encourage, the use of passphrases, and do not prevent password managers, long passphrases or highly complex passwords being entered.
- Verify all account identity authentication functions (such as update profile, forgot password, disabled / lost token, help desk or IVR) that might regain access to the account are at least as resistant to attack as the primary authentication mechanism.
- Verify that the changing password functionality includes the old password, the new password, and a password confirmation.
- Verify that any existing session identifiers prior to authorizing a new user session.
- Verify that an expiration date is set for sessions/credentials.
- Verify that all authentication decisions can be logged, without storing sensitive session identifiers or passwords. This should include requests with relevant metadata needed for security investigations.
- Verify that account passwords are one way hashed with a salt, and there is sufficient work factor to defeat brute force and password hash recovery attacks.
- Verify that credentials are transported using a suitable encrypted link and that all pages/functions that require a user to enter credentials are done so using an encrypted link.
- Verify that the forgotten password function and other recovery paths do not reveal information to recover the password or the current password itself and that the new password is not sent in clear text to the user.
- Verify that information enumeration is not possible via login, password reset, or forgot account functionality.
- Verify there are no default passwords in use for the application framework or any components used by the application (such as "*admin/password*").
- Verify that anti-automation is in place to prevent breached credential testing, brute forcing, and account lockout attacks.
- Verify that all authentication credentials for accessing services external to the application are encrypted and stored in a protected location.
- Verify that forgotten password and other recovery paths use a TOTP (Time-based One-time Password) or other soft token, mobile push, or other offline recovery mechanism. Use of a random value in an e-mail or SMS should be a last resort and is known weak.

- Verify that account lockout is divided into soft and hard lock status, and these are not mutually exclusive. If an account is temporarily soft locked out due to a brute force attack, this should not reset the hard lock status.
- Verify that if shared knowledge based questions (also known as "secret questions") are required, the questions do not violate privacy laws and are sufficiently strong to protect accounts from malicious recovery.
- Verify that the system can be configured to disallow the use of a configurable number of previous passwords.
- Verify that risk based re-authentication, two factor or transaction signing is in place for high value transactions.
- Verify that measures are in place to block the use of commonly chosen passwords and weak passphrases.
- Verify that all authentication challenges, whether successful or failed, should respond in the same average response time.
- Verify that secrets, API keys, and passwords are not included in the source code, or online source code repositories.
- Verify that if an application allows users to authenticate, they can authenticate using two-factor authentication or other strong authentication, or any similar scheme that provides protection against username + password disclosure.
- Verify that administrative interfaces are not accessible to untrusted parties.
- Browser autocomplete, and integration with password managers are permitted unless prohibited by risk based policy.

Sensitive Data Exposure.

- Classify data processed, stored or transmitted by an application. Identify which data is sensitive according to privacy laws, regulatory requirements, or business needs.
- Apply controls as per the classification.
- Don’t store sensitive data unnecessarily. Discard it as soon as possible or use PCI DSS (Payment Card Industry Data Security Standard) compliant tokenization or even truncation. Data that is not retained cannot be stolen.
- Make sure to encrypt all sensitive data at rest.
- Ensure up-to-date and currently considered to be strong by experts in the field, standard algorithms, protocols, and keys are in place; use proper key management.
- Encrypt all data in transit with secure protocols such as TLS with perfect forward secrecy (PFS) ciphers, cipher prioritization by the server, and secure parameters. Enforce encryption using directives like HTTP Strict Transport Security (HSTS).
- Disable caching for responses that contain sensitive data.
- Store passwords using strong adaptive and salted hashing functions with a work factor (delay factor), such as Argon2, scrypt, bcrypt or PBKDF2.
- Verify independently the effectiveness of configuration and settings.
- Certificates should be carefully managed and checked to assure that data are encrypted with the intended owner's public key.

- If certificate pinning is being used, ensure that all relevant properties of the certificate are fully validated before the certificate is pinned, including the hostname.

XML External Entities (XXE).

- Whenever possible, use less complex data formats such as JSON, and avoiding serialization of sensitive data.
- Patch or upgrade all XML processors and libraries in use by the application or on the underlying operating system. Use dependency checkers. Update SOAP to SOAP 1.2 or higher.
- Disable XML external entity and DTD processing in all XML parsers in the application, as per the OWASP Cheat Sheet *'XXE Prevention'*.
- Implement positive ("*whitelisting*") server-side input validation, filtering, or sanitization to prevent hostile data within XML documents, headers, or nodes.
- Verify that XML or XSL file upload functionality validates incoming XML using XSD validation or similar.
- SAST tools can help detect XXE in source code, although manual code review is the best alternative in large, complex applications with many integrations.

Broken Access Control.

- Bypassing access control checks by modifying the URL, internal application state, or the HTML page, or simply using a custom API attack tool.
- Allowing the primary key to be changed to another's users record, permitting viewing or editing someone else's account.
- Elevation of privilege. Acting as a user without being logged in, or acting as an admin when logged in as a user.
- Metadata manipulation, such as replaying or tampering with a JSON Web Token (JWT) access control token or a cookie or hidden field manipulated to elevate privileges, or abusing JWT invalidation
- Verify the CORS configuration, misconfiguration allows unauthorized API access.
- Force browsing to authenticated pages as an unauthenticated user or to privileged pages as a standard user. Accessing API with missing access controls for POST, PUT and DELETE.
- Verify that for any security checks that are performed on the client side, ensure that these checks are duplicated on the server side.
- Verify that input validation is performed, check the "Injection" Risk for further detail.
- Verify very carefully the setting, management, and handling of privileges. Explicitly manage trust zones in the software.
- Verify all pages and resources by default require authentication except those specifically intended to be public (Principle of complete mediation).
- Verify all authentication controls are enforced on the server side.
- Apply appropriate access control authorizations for each access to all restricted URLs, scripts or files.

Security Misconfiguration.

- A repeatable hardening process that makes it fast and easy to deploy another environment that is properly locked down. Development, QA, and production environments should all be configured

identically, with different credentials used in each environment. This process should be automated to minimize the effort required to setup a new secure environment.

- A minimal platform without any unnecessary features, components, documentation, and samples. Remove or do not install unused features and frameworks.
- A task to review and update the configurations appropriate to all security notes, updates and patches as part of the patch management process (see A9:2017-Using Components with Known Vulnerabilities). In particular, review cloud storage permissions (e.g. S3 bucket permissions).
- A segmented application architecture that provides effective, secure separation between components or tenants, with segmentation, containerization, or cloud security groups (ACLs).
- Sending security directives to clients
- An automated process to verify the effectiveness of the configurations and settings in all environments.
- Ensure that error messages only contain minimal details that are useful to the intended audience, and nobody else.
- Handle exceptions internally and do not display errors containing potentially sensitive information to a user.
- Create default error pages or messages that do not leak any information.
- Restricting access to important directories or files by adopting a need to know requirement for both the document and server root, and turning off features such as Automatic Directory Listings that could expose private files and provide information that could be utilized by an attacker when formulating or conducting an attack.

Insecure Deserialization.

- Implementing integrity checks such as digital signatures on any serialized objects to prevent hostile object creation or data tampering.
- Enforcing strict type constraints during deserialization before object creation as the code typically expects a definable set of classes. Bypasses to this technique have been demonstrated, so reliance solely on this is not advisable.
- Isolating and running code that deserializes in low privilege environments when possible.
- Log deserialization exceptions and failures, such as where the incoming type is not the expected type, or the deserialization throws exceptions.
- Restricting or monitoring incoming and outgoing network connectivity from containers or servers that deserialize.
- Monitoring deserialization, alerting if a user deserializes constantly.

Using Components with Known Vulnerabilities.

- Remove unused dependencies, unnecessary features, components, files, and documentation.
- Continuously inventory the versions of both client-side and server-side components (e.g. frameworks, libraries) and their dependencies using tools like versions, *DependencyCheck*, *retire.js,* etc. Continuously monitor sources like CVE (Common Vulnerabilities and Exposures) and NVD (National Vulnerability Database) for vulnerabilities in the components. Use software composition

analysis tools to automate the process. Subscribe to email alerts for security vulnerabilities related to components you use.

- Only obtain components from official sources over secure links. Prefer signed packages to reduce the chance of including a modified, malicious component.
- Monitor for libraries and components that are unmaintained or do not create security patches for older versions. If patching is not possible, consider deploying a virtual patch to monitor, detect, or protect against the discovered issue.

Insufficient Logging & Monitoring.

- Ensure all login, access control failures, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts, and held for sufficient time to allow delayed forensic analysis.
- Ensure that logs are generated in a format that can be easily consumed by a centralized log management solution.
- Ensure high-value transactions have an audit trail with integrity controls to prevent tampering or deletion, such as append-only database tables or similar.
- Establish effective monitoring and alerting such that suspicious activities are detected and responded to in a timely fashion.
- Establish or adopt an incident response and recovery plan, such as NIST 800-61 rev 2 or later.

Once this exercise is conducted, it is due going back to the list presented in Section 2 and present the mitigation actions that M-Sec will take in order to reduce the risk rating of the threats considered. Table 17 sums up this work.

Table 17. M-Sec mitigation for Application related threats

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.App.1	Libraries and modules on which the application is reliant, can be compromised or replaced by malicious versions. (they can be affected by the same threats as the application itself)		3	Vulnerability Assessment
Thr.App.2	Other malicious agents can issue requests and data on behalf of the application.	Connected Care	15	Companion DB may mitigate some of the risks; the application will not know the keys, only the user will know it. Authentication mechanism.
Thr.App.3	Malicious agents may have read access to the data the application is processing, and results.	Park Guide, Connected Care, Smile City Report	15	Companion DB may mitigate some of the risks. Only authorized agents can access M-Sec sensitive data. It is encrypted/decrypted through the Companion DB. Authentication mechanism.

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.App.4	Malicious agents may have write access to the data the application is processing. Being able to change it and produce unpredicted states	Connected Care	15	With the Companion DB, only authorized agents can access M-Sec sensitive data. It is encrypted/decrypted through the Companion DB. Authentication mechanism.
Thr.App.5	Data sources may be replaced, feeding erroneous or malicious data into the system workflow. E.g: Buffer overflow; cross-site scripting; SQL injection; canonicalization	Connected Care, Smile City Report	3	With the Companion Database, the data is encrypted and linked to the blockchain, so it cannot be tampering. Authentication mechanism.
Thr.App.6	Compiled, binaries or bytecode of the application may be corrupted or maliciously altered for execution.		15	Memory Protection
Thr.App.7	Legit requests may have undesirable effects.		15	Vulnerability Assessment
Thr.App.8	The user may be convinced to perform actions that expose their data, or the application workflow (Social Engineering)		3	Security Learning
Thr.App.9	Stored Data may be compromised. Either the cryptographic keys are not secure enough; the algorithms, the storage container is compromised or there might be some issue in the whole workflow.	Park Guide, Connected Care, Smile City Report	5	With the Companion Database, the storage of the sensitive dat is in a different database, so they should compromise at least the two databases. Vulnerability Assessment
Thr.App.10	The user account is compromised. Either because the user has released, forgot, or shared her/his credentials, or because the account is meant to be shared amongst several users.	Park Guide, Smile City Report	15	Log Mechanism
Thr.App.11	The application may be compromised, because there is some extreme cases that are not considered, or certain assumptions make it susceptible to get to unstable states	MTSA	5	Vulnerability Assessment MTSA can adapt application logic to satisfy its requirements under the updated environmental assumption.
Thr.App.12	The application (or platform) does not provide log of the transactions and/or execution trace. Leaving potential attacks un accounted.	Park Guide, Connected Care, Smile City Report	5	The Companion DB provides some logs of interactions, but it is not its main purpose. Vulnerability assessment.
Thr.App.13	The application uses un registered communications (not known to the underlying platform) or without relation to the functioning of the app itself.		5	Vulnerability Assessment

THREAT #	DESCRIPTION	M-SEC ASSET	RISK RATING	COMMENTS / MITIGATION
Thr.App.14	The application does not use the appropriate authorization mechanisms, or these mechanisms can be easily circumvented	Connected Care	5	In the backend site, the Companion DB uses a smart contract in order to grant access to the sensitive data. Vulnerability assessment.
Thr.App.15	The application does not use the appropriate authentication mechanisms, or these mechanisms can be compromised (e.g.: key logger, un secured password storage or transmission, etc.)		5	Vulnerability Assessment
Thr.App.16	Vulnerabilities-flaws in smart contracts	Blockchain app / Smart contract	15	Flaws in smart contracts can cause unforeseen security breaches. Thorough lab testing before going into production. Continuous code review.
Thr.App.17	Under-optimized smart contracts	Blockchain app / Smart contract	9	Dead code, loop fusion, repeated computation can cause denial of service on the long run. Thorough lab testing before going into production. Continuous code review.
Thr.App.18	Transaction privacy leakage	Blockchain app / Smart contract	9	Once identity is revealed the whole history of transactions is exposed. Thorough lab testing before going into production. Continuous code review.
Thr.App.19	Misunderstanding of the agreement of applications.	All apps	3	Agreement should be easy and clear to understand.
Thr.App.20	Personal information and facial images are mistakenly uploaded in the marketplace and traded.	Marketplace	15	Need a check mechanism
Thr.App.21	Malicious agents may make fake transactions.	Marketplace	15	Ensure by blockchain technology
Thr.App.22	Malicious agents may upload face data.	Marketplace	15	Ensure by blockchain technology

End-to-end Risks Mitigation

To sum up the risk mitigation actions presented under the rest of concepts and related with the potential threats associated to end-to-end mentioned in Section 2, below there is a list of mitigation actions to put into place.

Availability

- Verify that the required software and hardware are properly designed and implemented in develop time.

- Find and report runtime errors of softwares and hardware in IoT devices.
- Enable the IoT communication platform to reject abnormal or malicious connection or communication request from IoT devices to protect the platform from deny-of-service attack

Confidentiality

- Enable end-to-end authentication between data publisher and subscriber(s) via the communication channel provided by the IoT communication platform. The publisher and subcribers can verify the each other. A 3rd party PKI outside the IoT communication platfrom is used.
- Enable negotiation of encrytion keys between publisher and subscriber(s) using the communication channel provided by the IoT communicatino platform.
- Ennable end-to-end encryption on the data from data publisher and subscriber. The data are encrypted so that the IoT communication platfrom and other unauthorized users cannot access the data. Standad encryption protocoles are interated.

Integrity

- Verify if the data are altered maliciously. A digital signature will be attached to each piece of data. The signature can be verified with the help of 3rd party KPI.
- Verify if previous data are repeatedly transmitted. A timestamp will be attached to each piece of data.

Privacy, GDPR, PIPA, Ethics related Risks Mitigation

Upon the summary of potential threats M-Sec may face in this context and since the focus of Task 5.3 is on providing a compliance assessment to privacy and data protection regulations, the consortium considered to offer here a mention into mitigation actions would produce an overlap. The work performed in WP3 follows a different approach than the one done in WP5: everything done in WP3 should go attached to the principles of privacy by design, and each use case must in the end comply with the Privacy Compliance that will be evaluated by the corresponding Data Protection Officer (DPO). Nevertheless, to avoid redundancy, readers are invited to check Deliverable 5.11 *"M-Sec GDPR compliance assessment report"* out and find there the activities applied in the diverse use cases to fulfil this alleviation of threat level in relation to privacy.

5. Conclusions

The work conducted during the preparation of this report mainly focuses on providing a risk assessment which at its essence is a systematic examination of a task, job or process that in this case, members of the M-Sec consortium carry out for the purpose of identifying the significant hazards, the risk of someone being harmed and deciding what further control measures must be taken to reduce the risk to an acceptable level. All in all, risk assessment is a general process for linking science to decision-making and it is important to take into account it is not a monolithic process or a single method.

In this report the consortium presents the threats analysis methodology applied, introducing which kind of model is applied and how, along with detailed lists of potential threats that may affect the different layers in the M-Sec architecture.

Several threats looming over the M-Sec framework have been distinguished when carrying out this exercise, affecting the layers the M-Sec framework is composed of and making it clear some of those threats could turn into a real relevant risk and may require a prompt action to alleviate them. Therefore, during the execution of the different M-Sec use cases, diverse mitigation activities will be put into effect aiming at lowering those threats probability and criticality, and thus making the risk negligible.

Some of these mitigation activities are somehow linked to the so-called privacy enhancing technologies, which have also received their share of attention and will play an important role in executing the project's use cases in a format that ensures the security and privacy of data and users.

Given that the work in Task 3.3 ends at this stage, the results of the application of the proposed mitigation activities will be provided in a future deliverable emanating from Work Package 4. This way, it will be possible to demonstrate how those actions that theoretically would be indicated to mitigate the aforementioned threats to the system present a real effect over real-life deployments, as represented by M-Sec use cases.

Annex

The document is accompanied by a spreadsheet which has been the main source for extracting the main lists of threats (split into categories, groups, etc.), as well as the primary working file which was used internally by the consortium to gather and extract details for each threat.

In the spreadsheet, the following tabs can be found:

- **[File History]:** This tab is being used for partners to track internally changes being made in the spreadsheet. Since this spreadsheet will keep being used during Y3 (to update the Risks Mitigation progress in the corresponding Threats tabs), it is expected for it to change in the future.
- **[Definitions]:** This includes a brief presentation of the several headings present in the Threats tabs.
- **[Criteria – CIA]:** In this tab appears a recap of the model designed to guide policies for information security within M-Sec, where the consortium applies the Confidentiality, Integrity and Availability approach. Section 2.3 of this report presents the information registered in this tab.
- **[Threats]:** This is the main outcome of the spreadsheet and the primary working area used during the elicitation and the analysis of the threats. A lot of the content that it provides is presented in Sections 2.4 and 4 although it is possible to acquire more details in the future (as it will be explained below).

The spreadsheet is a valuable tool from which information can be grouped and organised so as to be documented (or visualised) for other activities of the project. By using sorting, filtering and hiding columns, it is possible to track the results, status and progress of the threats analysis through various perspectives. For example, it is possible to acquire the Risks Mitigation Progress for each Sub-Group or Group of threats.

At this stage, the spreadsheet is considered almost finalised for its initial purpose, although it is a living document as well. The only section that will be kept "alive" until the late stages of the project will be the "Risk Rating Y3" sections in the different [Threats] tabs, so as to keep monitoring the overall progress towards the fulfilment of the mitigation activities during Y3 and supporting the project validation activities at the end of M-Sec. Sections related to mitigation activities may also be updated.