D4.7– INITIAL SECURITY THREAT AND ATTACK MODELS

Deliverable ID	**D4.7**
Deliverable Title	**Initial Security Threat and Attack Models**
Work Package	**WP4 – Models and algorithms for CPS Swarms**
Dissemination Level	**PUBLIC**
Version	**1.0**
Date	**2018-10-26**
Status	**Final**
Lead Editor	**SLAB**
Main Contributors	**Regina Krisztina Bíró (SLAB), Bálint József Jánvári (SLAB)**

Published by the CPSwarm Consortium

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 731946.

Document History

Version	Date	Author(s)	Description
0.1	2018-01-10	Regina Krisztina Bíró (SLAB)	First Draft with TOC
0.2	2018-04-04	Regina Krisztina Bíró (SLAB)	Added Chapters 2 and 3
0.3	2018-06-05	Regina Krisztina Bíró (SLAB)	Added asset categorization and first version of asset inventory
0.4	2018-08-07	Regina Krisztina Bíró (SLAB) Bálint József Jánvári (SLAB)	Refined assets and added CIA analysis
0.5	2018-09-19	Bálint József Jánvári (SLAB)	Rearranged document chapters
0.6	2018-10-15	Regina Krisztina Bíró (SLAB) Bálint József Jánvári (SLAB)	Imported attack trees and added description
0.7	2018-10-18	Regina Krisztina Bíró (SLAB) Bálint József Jánvári (SLAB)	Added risk assessment and countermeasures
0.8	2018-10-19	Regina Krisztina Bíró (SLAB) Bálint József Jánvári (SLAB)	Added introduction and future work chapters
0.9	2018-10-20	Regina Krisztina Bíró (SLAB)	Ready for internal peer review
1.0	2018-10-26	Regina Krisztina Bíró (SLAB)	Integrated comments and suggestions from LAKE and UNI-KLU

Internal Review History

Review Date	Reviewer	Summary of Comments
2018-10-24	Melanie Schranz (LAKE)	Approved with minor comments
2018-10-25	Midhat Jdeed (UNI-KLU)	Minor comments

Table of contents

Internal Review History 2
1 Introduction 5
1.1 Goal 5
1.2 Summary 5
2 Methodology 6
2.1 State of the art research on methodologies 6
2.1.1 STRIDE 6
2.1.2 DREAD 7
2.1.3 Trike 7
2.1.4 P.A.S.T.A. 7
2.1.5 VAST 7
2.1.6 Attack trees 8
2.1.7 Failure mode and effects analysis (FMEA) 9
2.1.8 Fault tree analysis (FTA) 9
2.2 Establishing our methodology 10
3 Case Studies 11
3.1 Initial case study 11
3.1.1 Regulations 11
3.1.2 Industry standards 12
4 Assets 15
4.1 Assumptions 15
4.2 Generic assets 15
4.3 Analysis of relevant assets 18
5 Attack trees 20
5.1 The anatomy of an attack tree 20
5.2 Attacker motivations 20
5.2.1 Sabotage mission 21
5.2.2 Cause financial damage to swarm operator 21
5.2.3 Cause physical harm to human beings or property 21
5.2.4 Disturb environment or bystanders 22
5.2.5 Steal swarm member 22
5.2.6 Steal sensitive data 22
5.3 Methods of compromise 23
5.3.1 Damage or destroy swarm member 23
5.3.2 Redirect swarm member 23
5.3.3 Take advantage of behaviour 23
5.3.4 Modify mission parameters 24
5.3.5 Eavesdrop on communications 24
5.3.6 Impersonate swarm member 24
5.3.7 Impersonate operator 25
5.3.8 Modify firmware 25

5.3.9 Compromise operator environment 26
6 Countermeasures 27
6.1 A study on attacks and their mitigations 27
6.2 Design considerations and embedded countermeasures 28
6.2.1 Hardware platform and firmware 28
6.2.2 Deployment infrastructure 29
6.2.3 Communications infrastructure 29
6.3 Summary of proposed countermeasures 30
7 Risk Assessment 31
7.1 Methodology 31
7.2 Attacking a swarm member vs. attacking the swarm 32
7.3 Risk assessment 33
7.4 Risk assessment with countermeasures 34
8 Future work 36
8.1 Safety 36
8.2 Applicability of assets on designated project use cases 36
8.3 Evaluation guidelines 36
8.4 Attack tree modelling in Modelio 36
9 Bibliography 37

1 Introduction

1.1 Goal

In order to help the users of the CPSwarm Workbench build secure products, this document endeavors to describe the threat landscape autonomous swarms face, with two main goals:

- To provide guidance for developers on security aspects when using the CPSwarm Workbench
- To aid the development of a workbench that acts as an enabler for security and safety features

The reader of this document, by the end, should have a solid, high-level understanding of the threats faced by an autonomous swarm.

1.2 Summary

In Chapter 2, we provide a brief overview of the different methodologies that are used today in the field of safety and security to describe threats and risks in a systematic way. The rationale for selecting and adjusting our own methodology is also described.

In Chapter 3, relevant regulations and standards are described. This will (in a later version of this document) be expanded with use-case specific case studies that helped us – and will help others – understand the threats swarms face.

In Chapter 4, we identify and categorize the assets that are relevant to an autonomous swarm. We also investigate the security relevant properties of each asset using the CIA triad.

In Chapter 5, attacker motivations and attack methods will be explored using attack trees, in order to understand the kind of attacks operators of swarms may face and how these attacks affect the assets we have previously identified.

In Chapter 6, we categorize the low-level attacks that have been discovered in order to define the scope of the countermeasures that need to be developed and to propose concrete countermeasures.

In Chapter 7, we assess the risk of each of these attacks, based on their likelihood and severity – both with and without the proposed countermeasures.

In Chapter 8, we describe the direction the final version of this document, will take and provide an overview of possible new directions to explore.

2 Methodology

Usually as one of the first steps of conducting a security analysis, threat modelling takes a look at the system from the attackers' perspective. Its goal is to identify high value assets, as well as potential vulnerabilities and threats. Over time, a number of methodologies have been developed to establish a systematic way to conduct security analyses, the most famous of which is STRIDE/DREAD, the words themselves being mnemonics for their respective categories. Using these methodologies, threats can be identified, risks can be assessed and decisions can be made on the development of countermeasures.

2.1 State of the art research on methodologies

This chapter summarizes the most widely used threat modelling methodologies both in the security and safety domain. Concerning security, the following methodologies are going to be discussed: STRIDE, DREAD, Trike, P.A.S.T.A. and VAST; as well as Attack Trees as a method to assist in the description of threats. For safety, the terminology is fault modelling as most threats to safety usually arise from component failure – here FMEA and FTA are going to be discussed.

2.1.1 STRIDE

The STRIDE [1] methodology aims to classify known threats according to the kinds of exploits used or motivation of the attacker. It was one of the first threat modelling approaches developed and published by Microsoft in 1999. The name, STRIDE is an acronym formed from the first letter of each of the following categories used by it for threat classification:

- **Spoofing identity:** "Identity spoofing" is a key risk for applications that have many users but provide a single execution context at the application and database level. In particular, users should not be able to become any other user or assume the attributes of another user.
- **Tampering with data:** Users can potentially change data delivered to them, return it, and thereby potentially manipulate client-side validation, GET and POST results, cookies, HTTP headers, and so forth. The application/product in question should not send data to the user, which are obtainable only from and/or within the application/product itself. It should also carefully check data received from the user and validate that it is valid and applicable before storing or using it.
- **Repudiation:** Users may dispute transactions if there is insufficient auditing or recordkeeping of their activity. For example, if a user says, "But I didn't transfer any money to this external account!", and it is not possible to track his/her activities through the application, then it is extremely likely that the transaction will have to be written off as a loss. This is usually tackled by non-repudiation controls, such as web access logs, audit trails, etc.
- **Information disclosure:** Users are rightfully wary of submitting private details to a system. If it is possible for an attacker to publicly reveal user data at large, whether anonymously or as an authorized user, there will be an immediate loss of confidence and a substantial period of reputation loss. Therefore, it is a must to include strong controls to prevent user ID tampering and abuse, particularly if a single context is used to run the entire application/product.
- **Denial of service:** Designers should be aware that their applications/products may be subject to a denial of service attack. Therefore, the use of expensive resources such as large files, complex calculations, heavy-duty searches, or long queries should be reserved for authenticated and authorized users, and not available to anonymous users.
- **Elevation of privilege:** If an application/product provides distinct user and administrative roles, then it is vital to ensure that the user cannot elevate his/her role to a higher privilege one. In particular, simply not displaying privileged role links is insufficient. Instead, all actions should be gated through an authorization matrix, to ensure that only the permitted roles can access privileged functionality.

2.1.2 DREAD

DREAD [2] is another classification scheme designed by Microsoft used to quantify, compare and prioritize the amount of risk induced by each evaluated threat. The name of the methodology is an acronym formed from the first letter of the following categories to evaluate the risk level of a given threat:

- Damage potential
- Reproducibility
- Exploitability
- Affected users
- Discoverability

The DREAD risk level is calculated by adding the scores from each of the above categories and the dividing it by 5 – to obtain an average. The scores range from 0 to 10 – 0 is the lowest meaning minimal or no impact induced by the threat and that it is extremely difficult to perform and scale; while 10 is the highest score and indicates that the threat is easy to perform/scale or has a severe/widespread impact.

2.1.3 Trike

The Trike [3] methodology focuses on using threat models as an input for risk management. The risk based approach Trike is mostly used by auditing teams where the threat models are based on a "requirements model". This establishes an acceptable level of risk defined by stakeholders attached to each asset class. Threats build upon these requirements and are then assigned a risk value.

2.1.4 P.A.S.T.A.

The Process for Attack Simulation and Threat Analysis (PASTA) [4] is a seven-step, risk-centric methodology. It aligns business objectives and technical requirements by providing a seven-step process which takes compliance issues and business analysis into account. The seven steps of the methodology are the following:

1. **Define objectives:** business objectives, security and compliance requirements
2. **Define technical scope:** define the boundaries of the technical environment and the underlying infrastructure, applications and software dependencies
3. **Application decomposition:** identify use cases, define application entry points and trust levels, identify actors, assets, services, roles and data sources
4. **Threat analysis:** probabilistic attack scenarios analysis, regression analysis on security events and threat intelligence correlation and analytics
5. **Vulnerability and weaknesses analysis:** queries of existing vulnerability reports, issues tracking, mapping threats to existing vulnerabilities, design flaw analysis using use and abuse cases, scorings (CVSS/CWSS) and Enumerations (CVEs/CWEs)
6. **Attack modelling:** attack surface analysis, attack tree development and attack library management, attack to vulnerability and exploit analysis using attack trees
7. **Risk and impact analysis:** qualify and quantify business impact, countermeasure identification and residual risk analysis, risk mitigation strategies

2.1.5 VAST

VAST [5] is an acronym for Visual, Agile and Simple Threat modelling. The unique underlying principle of this methodology is the scalability of the threat modelling process throughout the infrastructure and software development lifecycle, integrating threat modelling into an agile software development methodology.

2.1.6 Attack trees

The attack tree approach identifies possible threats with conceptual diagrams called attack trees consisting of one root node, internal nodes, and leaf nodes. From the bottom up, nodes are conditions which must be satisfied in order to make the direct parent node true. There are two types of parent nodes: *AND* and *OR* (see Figure 1). In the first case, every child node must be satisfied, while in the second case, one is enough. When the root node of an attack tree is satisfied, the attack is complete.

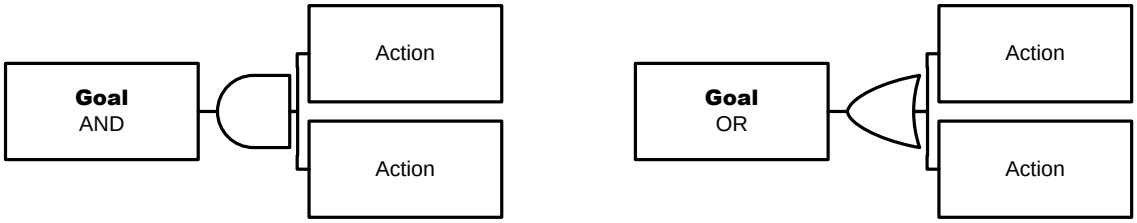


Figure 1 - AND and OR gates of an attack tree [6]

An example for attack trees is provided by the H2020 COSSIM project [6] which aimed to provide an integrated CPS simulation framework. Figure 2 depicts an attack tree describing an attacker's objective to obtain confidential data sent between the COSSIM Framework elements.

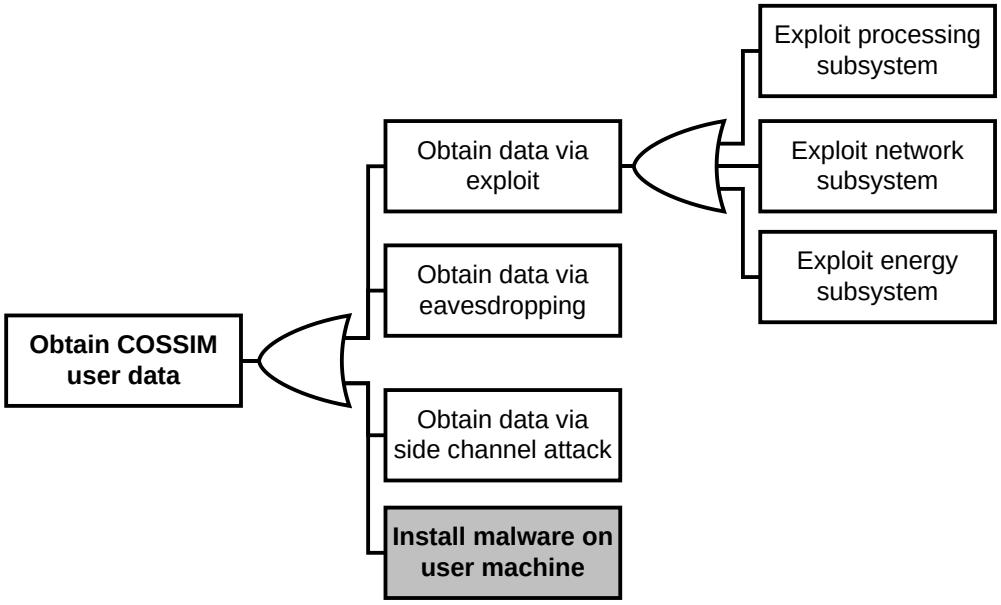


Figure 2 - Attack tree describing an attacker obtaining user data, from the COSSIM project [6]

The internal and child nodes represent sufficient attacks to reach the root node representing the goal of the adversary. It suggests that an attacker may be able to

1. obtain user data by exploiting a vulnerability in the COSSIM processing subsystem's implementation and read the data via direct memory access
2. perform eavesdropping on the communication between different components, and obtain data in that way as well
3. perform a side channel attack against the processing subsystem or energy subsystem
4. install malware on a COSSIM user's computer and steal confidential data from the user directly.

2.1.7 Failure mode and effects analysis (FMEA)

As a structured and systematic technique for failure analysis, FMEA has been in use for over 50 years to assess the reliability and safety of critical systems. Its main goal is to identify failure and eliminate (or at least minimize) the number of catastrophic failure conditions. It is an inductive process, as it considers a single failure at a time and examines its effect on the system as a whole. The analysis aims to identify and eliminate all single points of failure for the system, and should be conducted in parallel with the design process, to minimize the cost of developing countermeasures (see Figure 3).

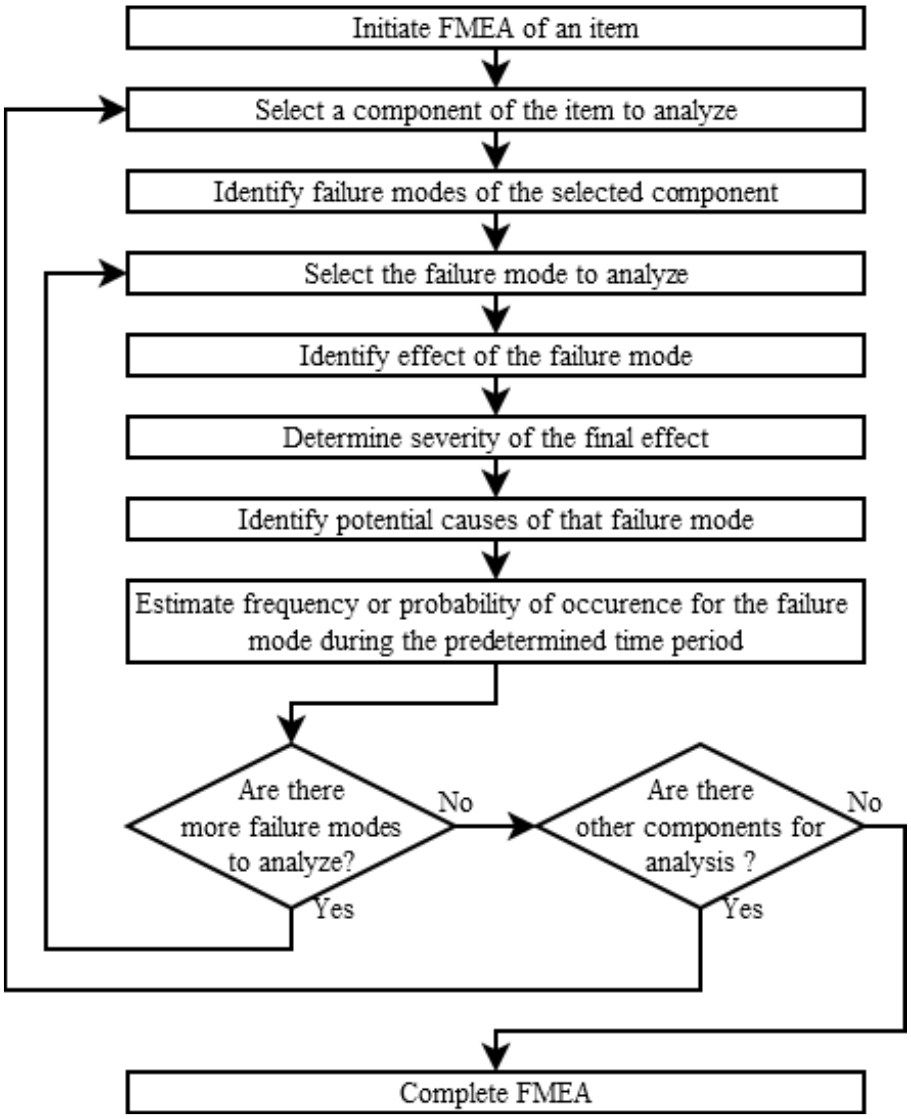


Figure 3 - FMEA flowchart [7]

2.1.8 Fault tree analysis (FTA)

Similar to attack trees, FTA builds an event tree for system failures (see Figure 4) using Boolean logic to combine fault indications from system components. Deductive reasoning is used to determine how different events contribute to a single system failure condition – starting from the top, the system state that needs to be avoided, and working backwards, trying to establish when that condition can occur. When events on which such a tree is built are combined with their probabilities, the tree itself can be used to ascertain the probability of system failure and to identify areas in need of additional countermeasures. A welcome side-effect of the analysis is that the resulting graph can also be used as a service manual for identifying the root cause of system problems.

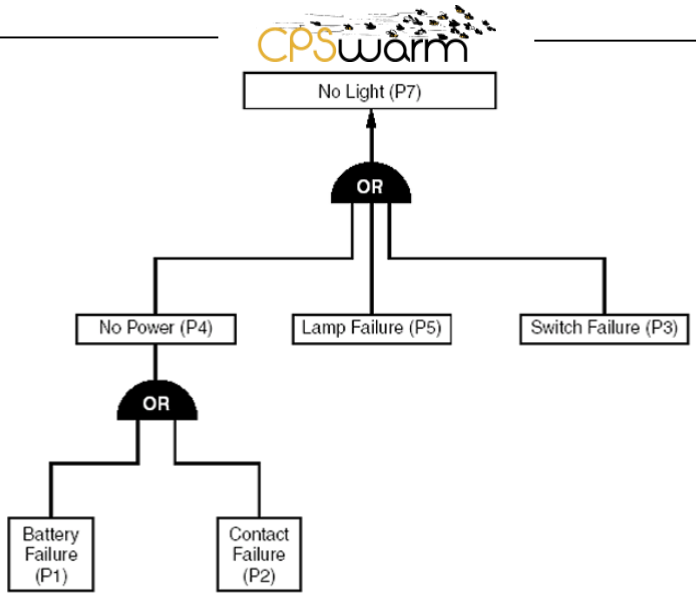


Figure 4 - Fault tree analysis on a vehicle headlamp [8]

2.2 Establishing our methodology

Since the CPSwarm project deals with CPSs which are connected with the physical world, it is desirable to be able to model both security and safety threats and faults. This is why Attack Trees and Fault Trees were chosen as tools for security and safety modeling. To get a unified solution – since the syntax and design elements are the same in both cases – SLAB will develop a plugin for Modelio for attack and fault tree analysis. This will ease the user workflow concerning safety and security as users can link their findings during the threat and fault analysis to existing security and safety solutions in the Modeling Library.

The threat modeling methodology, on the other hand, would not be taken directly from the above listed techniques – as most of these methodologies are designed for application security. However, since the CPSwarm project has three use cases and different stakeholder needs, this deliverable will follow the outline of P.A.S.T.A. – starting with establishing the scope of the work using a case study in the following chapter. Chapter 4 deals with defining generic assets for all use cases, Chapter 5, 6 and 7 will address the traditional threat models, risk assessment and countermeasures catalog. Case studies, assets, threat models, risk assessments and countermeasures specific for each of the three use cases will be added to the updated version of this deliverable, D4.8. Finally, the outline for future work regarding security evaluation, use case specific threat modelling and safety analysis will be described in Chapter 8.

3 Case Studies

3.1 Initial case study

This chapter presents a case study of industry standards, regulations and the current market landscape of products in the scope of the CPSwarm project, with the hope of inspiring the threat models and describing their scope. This deliverable only contains a non-use case specific case study, while case studies related to the specific use cases – where available - will be included in the next version of the deliverable (D4.8).

The chapter consists of two parts: the first is about regulations (mostly within the European Union) concerning the development and deployment of various Cyber Physical Systems related to the CPSwarm project. It is important to examine the regulations before conducting a threat analysis – they might provide insights on security and safety requirements that need to be addressed. The second part of the case study is of industry standards - following them sets a handful of requirements on safety and security.

3.1.1 Regulations

As of now, the European Union does not have specific legislation on robotics. Robots as products are regulated by a number of legislative frameworks, such as the Directive on Liability for Defective Products [9] and the Product Safety Directive [10]. To review the regulatory challenges posed by the advancing robotics technology, the Robolaw [11] project was funded under the European Commission's 7th Framework Programme for Research and Technological Development (FP7). The main objective of the Robolaw project was assessing whether existing regulations in the European Union are sufficient to address new problems brought by robotics technology and ensuring that the regulations provide conditions which incentivize European innovation in the robotics sector. Since the current and future regulations of robotics are complicated by the fact that there is no common understanding of what a robot is, the Robolaw project addressed this problem by identifying four categories where the application of the existing EU legislation would be problematic. These categories are

- Driverless vehicles
- Robotic prostheses
- Surgical robots
- Robot companions

Comparing differences and similarities of the above four categories, the Robolaw project proposed five main features with which robots can be categorized: autonomy, human-robot interaction, nature, environment and task. Based on these five features, the European Parliament has agreed on the characteristics that describe "smart robots":

1. Acquisition of autonomy through collecting data through sensors or exchanging data with its environment and analysing the data
2. Self-learning from experience or interaction
3. Having physical hardware components
4. Being able to adapt its behaviour and actions to dynamic environments
5. The absence of life in the biological sense

In the near future, the European Commission intends to analyse the above criteria and decide whether it is necessary for future regulatory purposes. Moreover, the Commission is planning to create definitions for three main categories within "smart robots", namely Cyber-Physical Systems, Autonomous Systems and Smart Autonomous Robots.

Since robotics technology is still a fairly new branch of industry, there are a number of upcoming regulatory and policy initiatives which are expected to be implemented by the European Commission:

- **Civil Law Liability:** the Commission will address legal questions related to the development and use of robotics and artificial intelligence in the next decade, and has already launched an evaluation of the Directive on Liability for Defective Products [12]. The extent to which the Directive can be applied to new technological developments, including advanced robotics and autonomous systems still needs to be evaluated.
- **Product Safety:** The Machinery Directive is currently being evaluated by the Commission to add better regulation principles. The revision may adapt the Directive's health and safety requirements to autonomous robots.
- **Autonomous cars and testing:** The Commission has launched several initiatives concerning autonomous cars, such as the European strategy on Cooperative Intelligent Transport Systems, connected and automated mobility (C-TIS) [13] and intends to establish cross-border testing corridors for these systems.
- **Harmonization of technical standards:** The are a number of research activities addressing the development of testing protocols for cooperative and collaborative systems which may lead to the creation of safety certification standards specific to the robots subject to these research projects.
- **An Advisory Body for Robotics and Artificial Intelligence:** the Commission proposes to create a high-level advisory body on robotics to advise the Commission.

Although there is a lot remaining to be decided, it is clearly foreseeable that the Commission's actions will significantly affect the development and research of robotics and artificial intelligence in the EU. The case studies in the updated version of this deliverable (D4.8) will collect current (and estimated future) requirements and regulatory initiatives which might concern the use cases of the CPSwarm project.

3.1.2 Industry standards

This chapter deals with the currently available security and safety standards in the robotics industry. The list may expand during the CPSwarm lifecycle as the industry advances – the goal is to provide an introduction to standards in robotics. Use case specific standards will be presented in D4.8.

3.1.2.1 ISO 12100, Safety of machinery - General principles for design - Risk assessment and risk reduction

The ISO 12100 standard specifies the basic terminology, principles and methodology for achieving safety in machinery design. It specifies risk assessment and risk reduction principles to help designers achieve their objectives. The standard is intended to be used as a basis for the preparation of

- type-B (generic) safety standards which deal with one safety aspect or one type of safeguard that can be used across a wide range of machinery and
- type-C (machine) safety standards dealing with detailed safety requirements for a particular machine or group of machines.

3.1.2.2 ISO 13849-1, Safety of machinery – Safety-related parts of control systems – Part 1: General principles for design

The ISO 13849-1 standard introduces required performance levels for safety-related control systems. The performance levels can be applied to the following safety-related parts of control systems:

- protective devices (e.g. two-hand control devices, interlocking devices), electro-sensitive protective devices (e.g. photoelectric barriers)
- control units (e.g. a logic unit for control functions, data processing, monitoring, etc.)

- power control elements (e.g. relays, valves, etc.)

3.1.2.3 IEC 62061, Safety of machinery - Functional safety of safety-related electrical, electronic and programmable electronic control systems

The IEC 62061 standard defines safety requirements for hardware and software and assigns safety integrity levels (SIL) for safety-related control systems (SRECS) as seen in Figure 5.

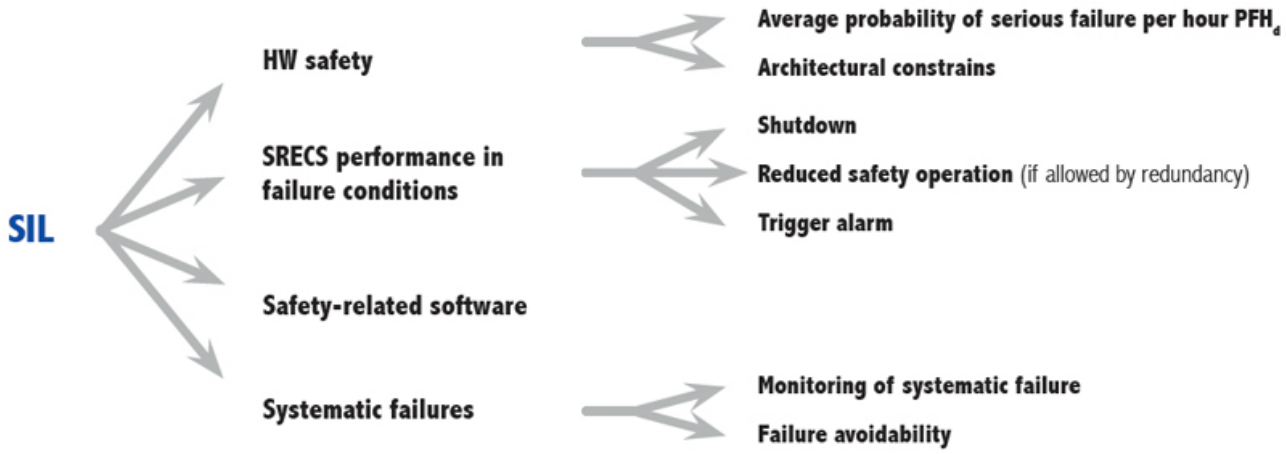


Figure 5 Safety Integrity Levels [14]

3.1.2.4 ISO 10218-1, Robots and robotic devices - Safety requirements for industrial robots -Part 1: Robots Requirements for the design of manipulators for industrial environments

The ISO 10218-1 standard specifies requirements and guidelines for the inherent safe design, protective measures and information for the use of industrial robots. It describes basic hazards associated with robots and provides requirements to eliminate, or adequately reduce, the risks associated with these hazards.

3.1.2.5 ISO 10218-2, Robots and robotic devices – Safety requirements for industrial robots – Part 2: Robot systems and integration

The ISO 10218-2 standard specifies safety requirements for the integration of industrial robots and industrial robot systems as defined in ISO 10218-1, and industrial robot cell(s). The integration includes the following:

- the design, manufacturing, installation, operation, maintenance and decommissioning of the industrial robot system or cell;
- necessary information for the design, manufacturing, installation, operation, maintenance and decommissioning of the industrial robot system or cell;
- component devices of the industrial robot system or cell.

Some examples include collaborative modes like monitored stop, hand guiding, velocity and/or force control.

3.1.2.6 ISO/TS 15066, Robots and robotic devices – Collaborative robots

The ISO/TS 15066 standard provides guidance for collaborative robot operation where a robot system and people share the same workspace. To achieve safety, robotic applications traditionally exclude operator access to the operations area while the robot is active. Therefore, a variety of operations requiring human intervention often cannot be automated using robot systems. In such operations, the integrity of the safety-related control

system is of major importance, particularly when process parameters such as speed and force are being controlled.

3.1.2.7 ISO 13482, Robots and robotic devices – Safety requirements for personal care robots

The ISO 13482 standard specifies requirements and guidelines for the inherently safe design, protective measures, and information for use of personal care robots, in particular the following three types of personal care robots:

- Mobile servant robots
- Physical assistant robots
- Person carrier robots

As a careful reader might have noticed, all of the above standards in robotics describe safety, but not cybersecurity. This is due to the fact that in the past cyber-security in the robotics industry was not a major issue, since robots and their controllers were not connected to the outside world in any meaningful way, let alone to the internet. However, the industry is changing and there is a push to connect these machines to the internet. The robotics industry has to adapt and maybe tailor ISO and ANSI standards on cybersecurity – which have been developed over decades and are already mature - in order to adjust industrial requirements for the era of Cyber-Physical Systems. In D4.8, this chapter will be extended with use-case relevant cybersecurity standards that could be used.

4 Assets

4.1 Assumptions

This chapter describes assets which are common for all of the use cases of the CPSwarm project. Use case specific assets will be described and identified in the next version of this deliverable, D4.8. First some common assumptions are presented which hold for all of the CPSwarm use-case scenarios and moreover, can cover other use cases of swarms of CPSs.

- A swarm of robots is deployed in a mission which requires information gathering and/or processing;
- Swarm members are equipped with different sensors, cameras and GPS modules;
- Swarm members can communicate with each other and/or the operator of the mission;
- The operator of the swarm initializes the mission by defining the objectives, targets and area of operation, and monitors the swarm remotely;
- External authorities such as police or border control may communicate remotely with the swarm in case of emergency or violation of local policies.

4.2 Generic assets

Having these assumptions in mind, we can now define the generic assets. Assets can be grouped into two categories: tangible and intangible assets. The intangible asset categories that should be protected when deploying a swarm of CPSs are

- *Information* - gathered and/or possessed by the swarm, including intellectual property and mission parameters,
- *Service* - meaning the capability to successfully execute the mission, includes the operability and performance characteristics of individual swarm members, and
- *Environment* - including safety and non-disruption to objects, humans and animals concerning the swarm's operation site.

We have identified the following primary intangible assets:

ID	Asset	Category
PA1	Operational parameters *Commands and additional information supplied by the operator to govern the behaviour of the swarm, including the area of deployment, the location of targets, etc.*	Information
PA2	Data gathered by swarm members *Any information collected by on-board sensors, including audiovisual feeds, component status and location, as well as information received from other swarm members or generated locally.*	Information
PA3	Swarm algorithm	Information

	The generic algorithm used by the swarm to solve the problem, which might be used to predict or sabotage swarm behaviour or might have significant market value.	
PA4	Presence of the swarm *The very fact that the swarm is operating in the vicinity.*	Information
PA5	Goal *The ability of the swarm to solve the specified problem and only the specified problem.*	Service
PA6	Controllability *The operator's ability to issue new commands, specify goals and in general maintain control over the swarm and its members.*	Service
PA7	Performance *The ability of the swarm to maintain the expected timeliness, manner and quality of service required to solve the task as expected by the operator, including the continuous feasibility of redeployment.*	Service
PA8	Environmental non-disturbance *The ability of the swarm to operate in a manner that does not disturb or interfere with the natural and manmade environment – operating such that no property damage occurs, the disturbance of bystanders is minimized and the natural environment is respected.*	Environment
PA9	Safety *The ability of the swarm to guarantee the safety of humans inside and outside its operational area.*	Environment
PA10	Compliance *The continuous assurance that the swarm and its members operate within the confines of the law and any applicable standards.*	Environment

The secondary, tangible assets are what we can protect - they support the primary, intangible assets and might possess vulnerabilities which can be exploited by an attacker aiming to corrupt the intangible assets - are the agents including their hardware and software components and operators including the personnel and the system they are using when setting up and monitoring the swarm. The tangible asset categories we would like to protect are

- *Software* – the operating system and software controlling the robots,
- *Hardware* – the body of the robot, including sensors and their perceived reality, and

- *Personnel* – the operators, developers and other privileged members of the organization responsible for the development and deployment of the swarm.

The protection of personnel assets is outside the scope of this document (and the project). With that said, the following secondary assets have been identified in the other categories:

ID	Asset	Category
SA1	Swarm member firmware *The operating system and embedded base software of individual swarm members that serve as the platform for the rest of the software.*	Software
SA2	Swarm algorithm implementation *The concrete implementation of the swarm algorithm as designed and deployed by the operator.*	Software
SA3	Operator software environment *The operating system and software that is present on the systems used by the operator.*	Software
SA4	Operator toolset *A set of software tools used by the operator to interact with the swarm and its members.*	Software
SA5	Communication protocol implementation *The implementation of the communication protocol used by either the operator toolset or the swarm algorithm implementation.*	Software
SA6	Locomotion *The ability of the CPS to move around at will in physical space.*	Hardware
SA7	Sensing *The hardware components responsible for gathering information about the environment and the CPS itself, including the correctness of such observations.*	Hardware
SA8	Actuation *The hardware components responsible for interacting with the environment.*	Hardware
SA9	Consumables *Any resource that is being consumed while the swarm is in operation – including battery charge, fuel, coolant, etc.*	Hardware

SA10	Processing *Computing resources – CPU time, RAM, etc. – that are required to execute the swarm algorithm and any dependent processes.*	Hardware
SA11	Structural integrity *The ability of the CPS to maintain its physical integrity throughout the mission.*	Hardware
SA12	Connectivity *Refers to the components that make up the physical layer of communications, including any radios, cabling, visual signage, etc.*	Hardware
SA13	Operator hardware environment *The underlying hardware that is used by the operator to run tools and control the swarm.*	Hardware

4.3 Analysis of relevant assets

One of the key concepts in information security is the CIA triad – Confidentiality, Integrity and Availability. We describe them below in terms of what they could mean in the context of a mission that uses swarms of CPSs.

1. *Confidentiality* means the non-disclosure of sensitive data to unauthorized parties – for instance the data collected or initially stored by the CPS, software and swarm algorithms and mission parameters.
2. *Integrity* means that sensitive data cannot be modified without authorization – for example data possessed by swarm members, software used in the runtime environment, mission parameters or emergency signals.
3. *Availability* means that the system, its components or certain functionalities must be available to operate when needed.

In this section we study whether the confidentiality, integrity and availability of each asset can be compromised. In Table 1 CIA analysis of assets we summarize the results according to the following legend:

- Y – yes
- N – no
- U – use case specific – will be expanded in D4.8
- N/A – not applicable

ID	Asset	C	I	A
PA1	Operational parameters	U	Y	Y
PA2	Data gathered by swarm members	U	Y	Y
PA3	Swarm algorithm	U	Y	Y
PA4	Presence of the swarm	U	N/A	N/A

PA5	Goal	U	Y	Y
PA6	Controllability	N/A	Y	Y
PA7	Performance	N/A	Y	Y
PA8	Environmental non-disturbance	N/A	N/A	Y
PA9	Safety	N/A	N/A	Y
PA10	Compliance	N/A	N/A	Y
SA1	Swarm member firmware	U	Y	Y
SA2	Swarm algorithm implementation	U	Y	Y
SA3	Operator software environment	U	Y	Y
SA4	Operator toolset	U	Y	Y
SA5	Communication protocol implementation	U	Y	Y
SA6	Locomotion	N/A	N/A	Y
SA7	Sensing	N/A	Y	Y
SA8	Actuation	N/A	N/A	Y
SA9	Consumables	N/A	N/A	Y
SA10	Processing	N/A	Y	Y
SA11	Structural integrity	N/A	N/A	Y
SA12	Connectivity	N/A	Y	Y
SA13	Operator hardware environment	N/A	Y	Y

Table 1 CIA analysis of assets

Confidentiality, where required, is always treated as use-case specific. In certain use cases, confidentiality can even be something that is explicitly forbidden or detrimental to the operation of the swarm member or the swarm. Nonetheless, where confidentiality is a potential requirement, the possible attack paths will be explored in this document.

5 Attack trees

5.1 The anatomy of an attack tree

In this section we briefly explain the visualization used for the attack trees using the Attack-Defense Tree Tool (ADTool) [15] which is an attack-defense tree visualization tool associated with the TREsPASS FP7 research project [16]. Although this tool is well aligned with the purpose of this deliverable, by the time of the release of the updated version of this deliverable, D4.8, SLAB will develop a Modelio plugin for drawing attack trees, making threat modelling an additional feature of the CPSwarm Workbench.

As already mentioned in Section 2.1.6, an attack tree is a hierarchical diagram consisting of one root node, internal nodes, and leaf nodes. From the bottom up, nodes are conditions which must be satisfied in order to make the direct parent node true. There are two types of parent nodes: *Type AND* and *Type OR*. In the first case, every child node must be satisfied; in the second case, at least one is required. In this chapter, we will denote the *AND* type node as seen in Figure 6 while the *OR* type parent nodes will be denoted as seen in Figure 7. When the root node of an attack tree is satisfied, the attack is complete.

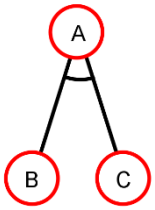


Figure 6 Example for the AND clause

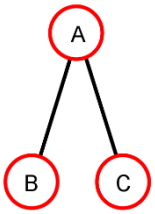


Figure 7 Example for the OR clause

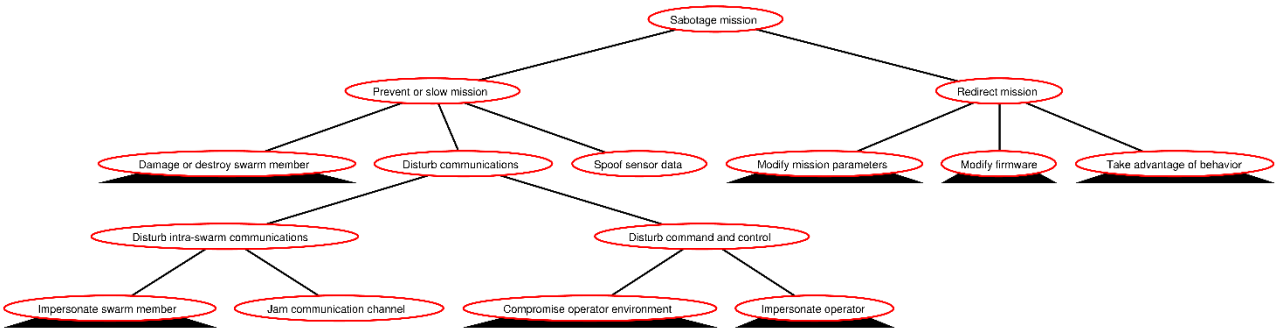
To enhance readability, we present smaller, narrower attack trees, thus we will detail specific attack types in separate trees, mostly in Section 5.3. In higher level attack trees, as in Section 5.2, we denote nodes which are expanded in a separate tree as seen in Figure 8.

Figure 8 Example for node expansion

5.2 Attacker motivations

This section presents the high-level attack trees that describe the possible goals of the different malicious actors we identified. Each of these will include an explanation of the motive and related attacks including the description of the affected primary assets.

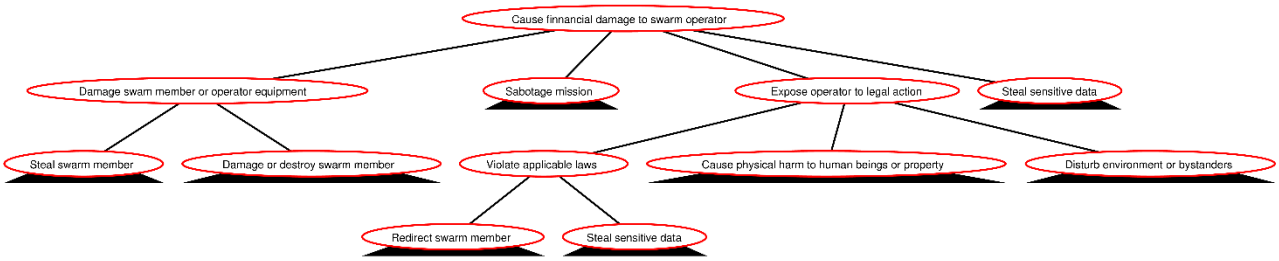
5.2.1 Sabotage mission



The reasons behind mission sabotage can span from a simple prank to a serious act of warfare resulting in the loss of human lives - depending on the application of the swarm in question. In D4.8, potential attacker motivations will be discussed for each of the use cases.

A sabotaged mission (either total or partial failure of it) can affect the majority of primary assets: PA1, PA2, PA3, PA5, PA6, PA7; while PA9 - Safety is affected based on the use case of the swarm, which will be assessed later. All the secondary assets can be affected depending on the application of the swarm and attack type.

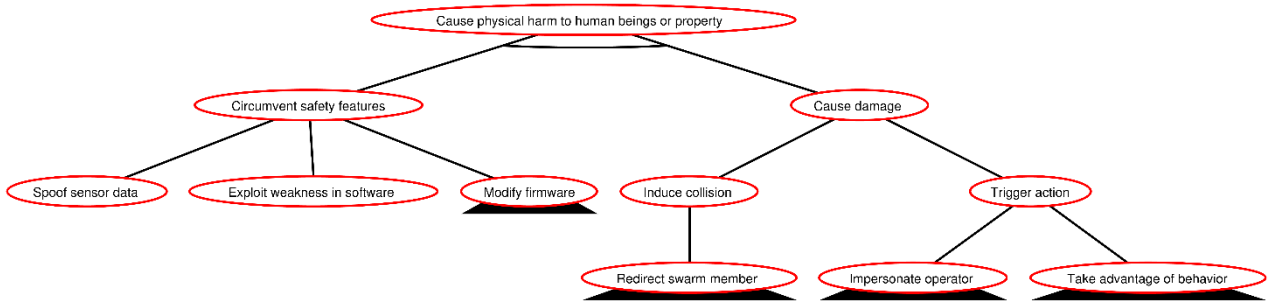
5.2.2 Cause financial damage to swarm operator



Financial damage to the swarm operator could be pursued by commercial competitors (to the operators) or criminals. Not only physical damage or exhaustion of resources can lead to financial damage, but also violation of local regulations by the swarm or the theft of intellectual property owned by the operators.

Here the primary assets affected are the ones not connected to physical damage – PA8, PA9 and PA10, while the damage of swarm members or equipment concerns several secondary assets – SA3, SA4, SA5, SA6, SA7, SA9, SA11 and SA13.

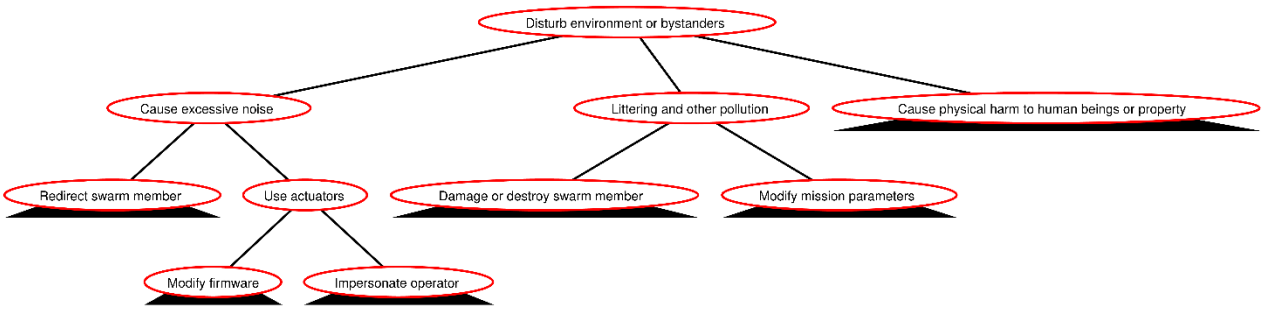
5.2.3 Cause physical harm to human beings or property



Vandals and criminal organizations may desire to cause physical harm to private or public property, while terrorist groups could corrupt swarms in order to take as many lives as possible and to cause mass destruction. To successfully perform this, they not only need to find a way to possibly cause damage but also have to figure out how to circumvent safety features applied to the swarm.

The affected primary assets are PA1, PA2, PA3, PA8 and PA9 while the secondary assets are SA1, SA2, SA3, SA4, SA7, SA8 and SA13.

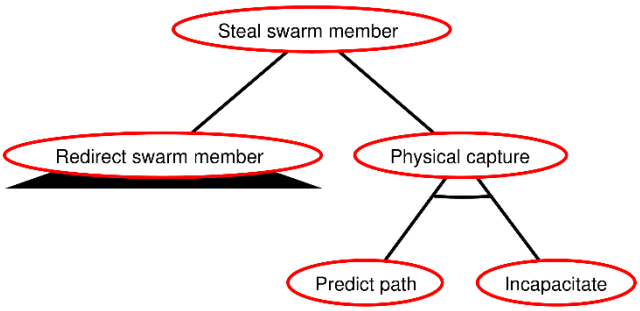
5.2.4 Disturb environment or bystanders



Vandals may want to disturb the environment or humans just for fun, while more serious criminals would want people or local authorities to shift their attention to the disturbance caused by the swarm while they can commit other crimes simultaneously, for example theft or robbery.

Means of destruction can also be used for distraction; hence the same primary and secondary assets are affected as in Section 5.2.3 (PA1, PA2, PA3, PA8, PA9, SA1, SA2, SA3, SA4, SA7, SA8 and SA13).

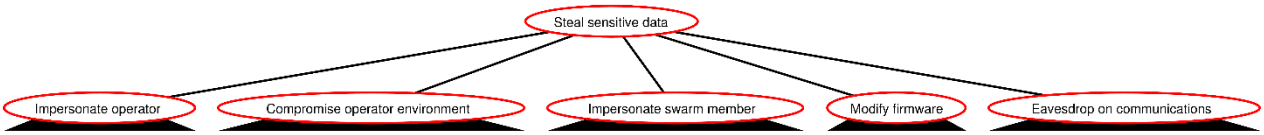
5.2.5 Steal swarm member



Theft could be done for profit by criminals or maybe by vandals as a form of self-entertainment. Since it requires physical interaction with the swarm members, a thief can either redirect swarm members to a preferred location or physically capture them – we have seen lots of different ways of doing that in the media lately, such as throwing nets or shooting blunt objects.

All primary and secondary assets are affected which can be associated with the physical entity of swarm members: PA1, PA2, PA3, PA6, SA1, SA2, SA3, SA4, SA5, SA6, SA7, SA8, SA9, SA10, SA11, SA12 and SA13.

5.2.6 Steal sensitive data



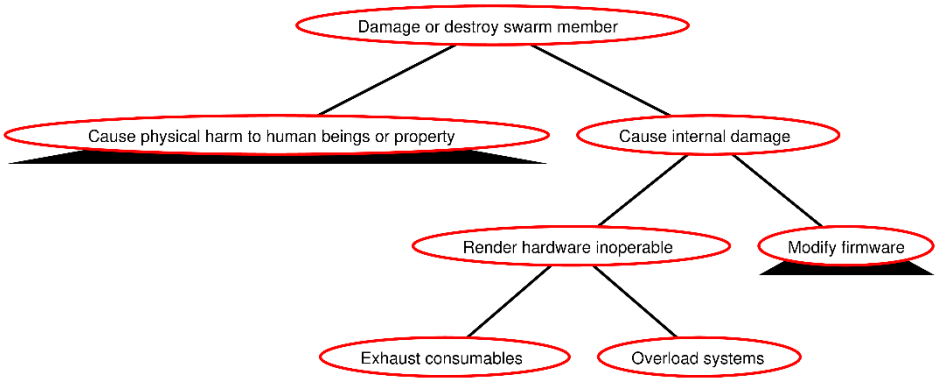
Sensitive data could be any data collected or possessed by the swarm which has to be kept secret. Similarly to stealing the physical devices, theft of sensitive data could be done for profit by criminals or by vandals for fun. In extreme cases, commercial competitors to the operators may want steal secrets connected to the swarm algorithms used or other intellectual property.

Since this is a high-level attack tree, all the assets that are affected by the attacks described by the leaf nodes are affected here as well – all primary and secondary assets may be affected based on what is regarded a secret in the mission of the swarm in question.

5.3 Methods of compromise

This section presents the lower-level attack trees that describe the possible actions an attacker has to accomplish to realize an attack. As seen in the previous section, a combination or selection of these could lead the attackers to realize their goal described above.

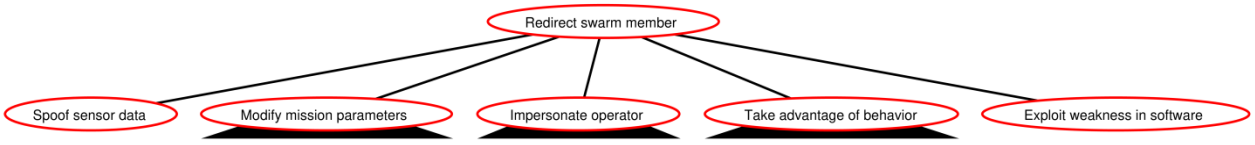
5.3.1 Damage or destroy swarm member



Damaging a swarm member is both a way to cause property damage and to diminish the efficacy of the swarm. Apart from attacks that try to damage the property of others (and also involve damage to swarm member itself), the swarm member might be rendered inoperable without it ever being visible on the outside – either by bricking the software environment or by causing excessive wear on parts.

Secondary hardware assets are the main targets of this attack (SA6, SA7, SA8, SA9, SA10 and SA11). Depending on the scope of the attack, the Goal and the Performance of the swarm might be diminished (PA5 and PA7).

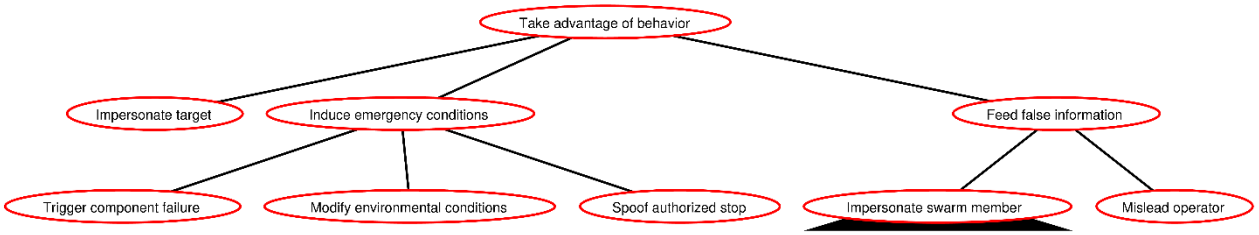
5.3.2 Redirect swarm member



For many attacks, the attacker needs to move the swarm member to a desired location – which can be achieved by traditional attacks that target the command and control infrastructure, or through more creative means, which try to take advantage of the behaviour of the swarm member by spoofing targets, current location and other sensor data to get the swarm member to move to the right place at the right time on its own.

This affects the primary assets in the Service category (PA5, PA6 and PA7), as well as this in the Environment category (PA8, PA9 and PA10). Of the secondary assets, Locomotion is targeted (SA6), but an attack may also affect others.

5.3.3 Take advantage of behaviour

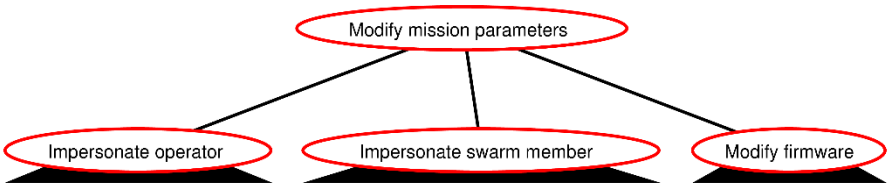


If the behaviour of the swarm member follows a known pattern, an attacker can use that to its advantage by manipulating the environment in order to get the swarm to behave differently. This might include spoofing targets by reverse engineering the method used to identify them or feeding entirely false information either

through the communication protocol or by providing misleading information to the operator. In certain cases, the attacker might also be interested in triggering an emergency condition – which may lead to the swarm member deactivating.

This attack – depending on how it is used – can affect a great variety of assets. The Swarm Algorithm (PA3) needs to be already compromised if this attack is to be successfully mounted.

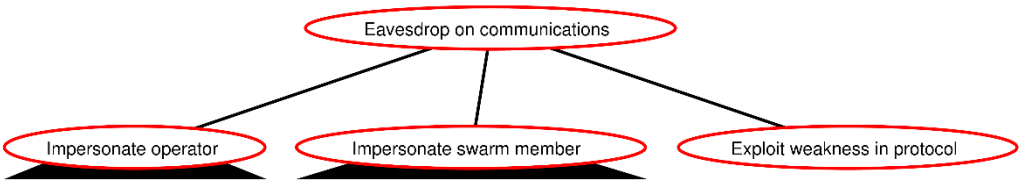
5.3.4 Modify mission parameters



Mission parameters are set by the operator to define the boundary conditions under which the swarm operates. These might include the location of targets, the operational area or the distance to keep from dangerous objects. Since mission parameters might be updated on-the-fly, or might be changed as a result of the information provided by other swarm members, any compromise of the communication infrastructure can potentially lead to these parameters being changed.

Modifying mission parameters can seriously affect the Goal and the Controllability of the swarm (PA5 and PA6), but might have consequences for other Service and Environment assets.

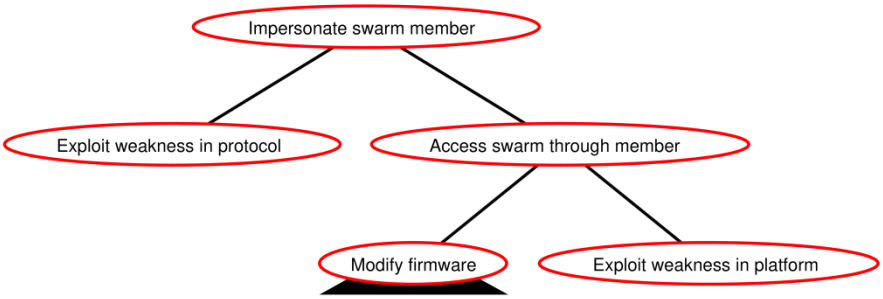
5.3.5 Eavesdrop on communications



Eavesdropping requires a physical compromise of the underlying communication medium. Since in most cases, this refers to receiving radio signals, for which (depending on the exact protocol and technology) commercial equipment is widely available, the tree does not deal with gaining access to the medium itself. Performing this attack relies on the attacker's ability to either find a weakness in the protocol or to successfully act as a member or operator of the swarm. The term protocol refers to both the high level protocol used by the swarm and any other protocols the communication stack uses.

A successful attack affects the confidentiality of all primary assets in the Information category (PA1, PA2, PA3, and PA4) and the Goal itself (PA5) – and of relevant secondary assets that might be transmitted over the link (SA1 and SA2).

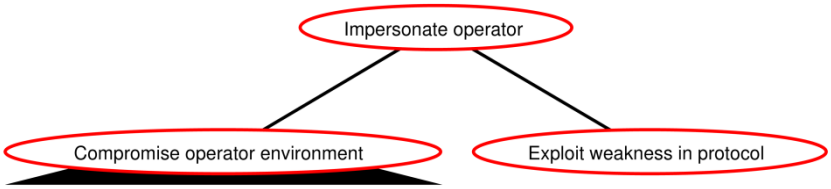
5.3.6 Impersonate swarm member



To successfully impersonate a swarm member, the attacker has to be able to send and receive messages in a way that it is indistinguishable from another existing (or a newly introduced, non-existent yet nonetheless recognized and accepted) member of the swarm. If a weakness in the protocol is found, the attack can even be performed without interacting with the swarm member in question – but a more likely scenario is that a swarm member is compromised in order to send and receive messages in its name.

Such an attack affects the confidentiality of any information shared between swarm members (and might, in most cases, imply 0), and can also compromise the integrity of information as received by the operator or other members (PA2).

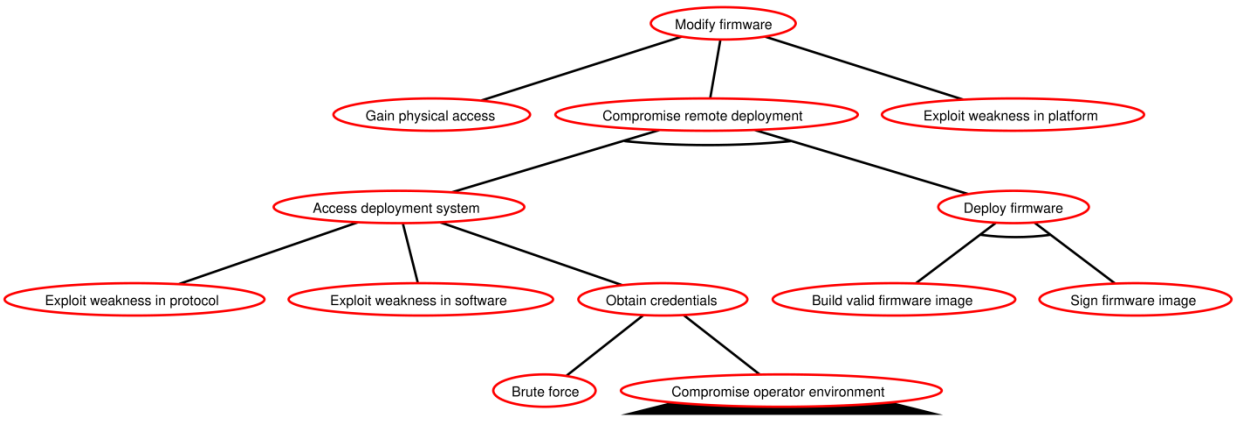
5.3.7 Impersonate operator



A significantly more dangerous attack than simply impersonating another swarm member is to impersonate the operator itself. This assumes that there is a separation of privileges – and that there are certain, often dangerous operations that can only be performed by the operator. If a weakness in the protocol is found, no interaction with the real operator is necessary – otherwise, the attacker has to first compromise the operator environment.

If successful, very little remains off the table for the attacker – the attack certainly implies a successful 0, as well as a total compromise of the availability and integrity of primary assets in the Service and Environment category (PA5, PA6, PA7, PA8, PA9 and PA10). Secondary assets are similarly affected.

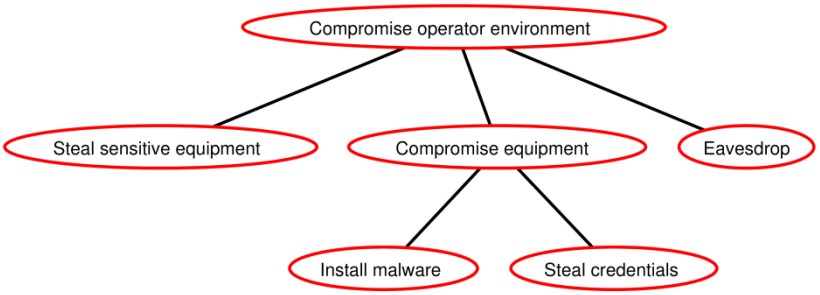
5.3.8 Modify firmware



Like traditional computing systems, swarm members are also subject to software modifications resulting from physical access or from exploits affecting the underlying operating system. In our model, however, swarm members are also subject to software updates through a remote deployment system. As it is standard practice with software updates, the privilege of installing updates and the privilege of authorizing the update and certifying its authenticity is separated – the attacker has to compromise the deployment system and then deploy an integrity protected update that the system recognizes as valid.

Modifying the firmware implies a total takeover of one swarm member – or if the attack is repeatable, most likely all members. Its effects can be similar to that of 5.3.6 or even 5.3.7, depending on the scope of the changes.

5.3.9 Compromise operator environment



The weakest link in most systems is the human – and for our swarm, this relates to the operator. The operator environment includes the computers, software and network connection used by the operator, as well as any information stored on its premises. By compromising this environment, the attacker can gain access to the system without the need to exploit any weaknesses in the swarm itself.

As this is a supporting attack, its effects on assets depend on the information obtained and on how that information is used. In the worst case, it can lead to a total takeover as in 5.3.7.

6 Countermeasures

6.1 A study on attacks and their mitigations

In Chapter 5 we have identified attacker motivations and then lower level attacks that can corrupt assets corresponding to a swarm in mission. In Table 2 we collected the low level attacks that can be mitigated. We have analysed whether they are in scope of the CPSwarm project – for example, physical security, social engineering and generic information security does not concern the main goal of the project and thus we will not include countermeasures or analyses regarding these. Some attacks can be carried out in many different ways depending on the application scenario and the type of CPSs used in the swarm, hence different countermeasures may apply to these. The countermeasures concerning these will be described in D4.8.

Attacks	In scope?	Type	Countermeasures
Brute force credentials	Yes	Communication	See next chapters
Exploit weakness in protocol	Yes		
Jam communication channel	Yes		
Spoof authorized stop	Yes		
Exploit weakness in platform	Yes	Hardening	
Sign firmware image	Yes	Deployment	
Impersonate target	Yes	Use case specific	Will be addressed in D4.8
Predict path	Yes		
Exhaust consumables	Yes		
Spoof sensor data	Yes		
Exploit weakness in software	Yes		
Build valid firmware image	No	Obscurity	N/A
Gain physical access	No	Physical attacks	
Incapacitate swarm member	No		
Modify environmental conditions	No		
Steal sensitive equipment	No		
Trigger component failure	No		
Mislead operator	No	Social engineering	
Install malware on operator environment	No	Generic information security	
Steal operator credentials	No		

6.2 Design considerations and embedded countermeasures

The CPSwarm project is focused on building tools that aid the development of autonomous CPS swarms. The final goal of exploring the threat landscape from the perspective of future users and operators is both to help us make better, more capable tools that have been designed with security in mind and to help these future users use these tools to build secure swarms.

While security analysis will have to be performed for each use case independently, unsurprisingly the attacks described so far point to a few major entry points:

- Hardware platform and firmware
- Deployment infrastructure
- Communications infrastructure
- Operator environment

A remote attacker has two choices: mount an attack against swarm members or mount an attack against the operator. We will not tackle the latter – it is the responsibility of the operator to properly secure its systems, both physically and from an IT security perspective. The former, however, is at the core of our project goals, and as such, the three relevant main areas where attacks are expected to happen need to be covered.

6.2.1 Hardware platform and firmware

The security of any software product depends heavily on the security of the underlying platform - no matter how carefully the developer builds the software, the system is likely to be compromised if the platform itself is vulnerable. The field of robotics has traditionally been a world of closed systems, with robots working without any network connection or only communicating on a local, closed network with their peers. Attacks against such systems often had to rely on the human element, like spreading malware through USB drives. As industries work toward increased connectivity, more and more devices are placed on public networks or networks where bridges exist to a public network – and as new attack surfaces open, the robotics industry now faces the challenge of securing these devices in the face of remote attackers.

The main robotics platform used in the CPSwarm project, ROS (Robot Operating System), is a Linux-based system with a number of custom packages and its own IPC system. Its defaults are completely insecure – most stock firmware images contain no security features whatsoever, the ROS communication model is devoid of any authentication or authorization scheme. While ROS2 is under development, and will eventually try to address some of these issues, it is not yet production ready and lacks the software and hardware ecosystem that was built around its predecessor. As such, if ROS-based devices are connected to the internet, they might be vulnerable to a variety of attacks even if no additional custom software is being used.

To set up the hardware and software platform, the following generic steps need to be taken:

1. First, **the platform needs to be updated**. For production environments – with the CPS having planned lifetimes measured in years – this includes using software that will be supported with security updates for the foreseeable future.
2. The second step is to configure the platform correctly and to **use only features that are inherently secure or not security sensitive**. What constitutes a correct configuration depends on the use case.
3. The third step is related to how the system is customized for the underlying hardware. **Sensors and actuators need to have proper error handling, input and output validation** – including sanity checks. Any communication equipment used must be set up to take advantage of low level security features provided by the physical layer.
4. Lastly, **settings need to be validated and saved** – to have a known starting configuration the system can be restored to. Follow up work also includes tracking and installing security updates.

These setup and maintenance steps are required to provide a stable platform for both swarm-related and other activities, but they are especially important for connected swarm applications where attack surfaces are significantly larger.

6.2.2 Deployment infrastructure

In this context, the deployment infrastructure consists of all components supporting the remote installation and update of software components and configuration. This includes the Deployment Tool as developed in the CPSwarm project, but might also include other third party software components used by the operator. Since the basic premise of these components is that they allow the operator to change the software running on the CPS, if this system is compromised, the attacker can, in the worst case, compromise the swarm completely.

Since deployment is performed remotely, the **communications infrastructure is also involved**, and all the remarks and countermeasures described there also apply. Whether the deployment infrastructure should share the authentication scheme used by other communications should be evaluated on a case-by-case basis, but from a security perspective, a complete separation is a better option. In any case, authentication and authorization should only be the first line of defence.

Another layer of protection can be applied to the deployed artefacts themselves – by **signing all packages and validating their signature** before deployment on the device, even if an attacker can get through whatever authentication measures are in place, he won't be able to deploy arbitrary packages without also compromising the private key used by the operator to issue the signed packages. This also allows a separation of privileges between the personnel performing the deployment and the people responsible for the development and the issuance of valid software packages – in certain commercial applications, these groups might belong to different departments or even companies.

Even if the attacker gains access to both the deployment infrastructure itself and can produce a properly signed package, one last line of defence can be established by limiting the scope and privileges of the packages being deployed. While for simple applications, the package might be a full-fledged firmware image, in which case its compromise would lead to a total takeover, for most high complexity system the deployment of new behaviour would be limited to the deployment of binaries and configuration files. In such a case, **industry standard isolation techniques** – limiting capabilities and containerization, file system isolation and so on – can be used to limit the damage that can be done by any deployed package. If this isolation is established correctly, the operator would still be able to shut down the rogue CPS, preventing further damage. Any such isolation must also include elements that limit behaviour to a safe range, checking the sanity of the input that is received from whatever control algorithm is deployed on the CPS. Ideally, any safety critical functionality would be protected by the isolation.

6.2.3 Communications infrastructure

While it is possible to use swarm algorithms that do not require direct communication between members (instead relying on sensory input), even such applications will likely communicate with the operator or the environment. This connectedness – as already mentioned in 6.2.1 – is the root of many security problems that plague modern robotics.

At the very least any communication scheme utilized either must implement some form of **authentication** to limit participation in swarm communications. An alternative would be to limit the scope of communication and the actions that could be remotely performed using these facilities – but even that would open up a significant attack surface.

The authentication scheme developed must be combined with the **strong integrity protection of messages**, so that for any message the recipient can determine whether it was sent by another member of the swarm (or the operator). In most use cases, it is also important to determine which swam member sent a particular message, and to be able to revoke access from compromised swarm members. This can be achieved using public key cryptography and by maintaining a chain of trust rooted at the operator. Provisioning devices at the time of their first use with certificates and establishing the trust relationship between the swarm and its new

member is an important first step that must be performed in a secure environment, since before that trust relationship is established, no secure remote communications can take place.

Building on a working authentication scheme combined with the strong integrity protection of messages, one can extend the scheme to cover **authorization**. In the context of swarm intelligences, this ensures that the operator is in a privileged position to issue certain commands that other parties in the communication – ordinary swarm members, IoT devices – cannot issue.

Authorization alone is insufficient to limit access to sensitive data, as once an authorized party requests such information others may eavesdrop – the solution is to protect the confidentiality of the messages using **encryption**. The need for encryption is use case dependent, and in certain cases might prove to be problematic – especially where performance and latency requirements make it impossible to use. Selective encryption and prioritization of messages is a possible solution – confidential telemetry data can usually tolerate higher latencies and jitter than high priority, safety critical messages (which might have no confidentiality requirement at all).

Wherever possible, industry standard technologies should be used – including for the physical layer of communications, which might bear some of the burden these countermeasures place on the implementer. Proving the correctness of any security relevant protocol is no easy task, and any concrete implementation of a protocol with these features will also be subject to attacks against the implementation itself.

6.3 Summary of proposed countermeasures

Any real world application of swarms should, at the very least:

- Build on up-to-date, supported, correctly configured platforms
- Implement security and safety critical functionality isolated from the main behaviour
- Use authenticated communications facilities, with dangerous actions requiring authorization
- Protect the confidentiality of sensitive communications with strong cryptography
- Implement remote deployment with multiple layers of security and isolation, or not at all
- Evaluate the impact of security features on the performance of the swarm

Within the CPSwarm project, a platform is being built that enables and empowers developers to achieve these goals and more. It is not on this project alone to supply all pieces of the puzzle – but the pieces the project supplies must help and not hinder these goals.

7 Risk Assessment

This chapter describes a risk assessment based on the identified threats and attacks and their effect on the assets identified in Chapter 4. In this deliverable only a non-use case specific risk assessment with the generic threats and assets identified will be included; use case specific risk assessment will be included in the next version of the deliverable (D4.8).

7.1 Methodology

A risk assessment is the combination of the following two procedures:

1. Identifying and analyzing potential events that may negatively impact assets and
2. Evaluating the risk: making judgements on the tolerability of the identified risks while considering the influencing factors.

In short, a risk assessment analyzes what can go wrong, how likely it is to happen, what the potential consequences are and how tolerable the risk is.

In Chapter 4 we have already identified the assets we want to protect, while in Chapter 5 we have already described potential events and attacks in sections 5.2 and 5.3 respectively, by using attack trees. What remains to complete the risk assessment are the definitions of the metrics based on which we can determine the likelihood and severity of threats to identify risks. When evaluating the likelihood, we will be using a qualitative approach. The likelihood scale and the interpretation for the levels are presented in Table 3.

Likelihood	Qualitative interpretation
3: Certain	There is a high chance that the scenario successfully occurs in a short time
2: Likely	There is a high chance that the scenario successfully occurs during the life time of the application of the swarm
1: Unlikely	There is little or no chance that the scenario successfully occurs in a short time

Table 3 Likelihood evaluation

To determine the severity of an attack, we use the following three levels:

- *Low* (1): Indirect or negligible attacks on the swarm fall into this category. The attacker can also obtain access to information, which may help executing other attacks against the swarm or the operators.
- *Medium* (2): The attacker can access sensitive information, data collected by swarm members, mission related parameters; or can cause persistent delays in the mission. The confidentiality and/or integrity of swarm data is endangered by the attacker.
- *High* (3)*:* The attacker can obtain control of the swarm, or can cause permanent damage to the swarm, humans or the environment.

On Table 4, the risk levels are estimated from the likelihood of attacks and their severity in a risk matrix. The risk value can take the following levels:

- *High*: the threat significantly endangers related assets
- *Medium*: the threat has a noticeable effect on the security of the related assets
- *Low*: The threat has a minor effect on the security of the related assets

To make the following table better readable, we assign different colors to different risks: high –red, medium – yellow and low – green. For instance, a likely accident with high (3) severity has a high level risk.

	Severity		
Likelihood	1	2	3
3: Certain	Medium	High	High
2: Likely	Low	Medium	High
1: Unlikely	Low	Low	Medium

Table 4 Risk matrix

Now that we have set all the metrics to evaluate the risks, in the following sections we determine the severity and likelihood of the low-level attacks described in Section 5.3 first. Then we take the attack scenarios presented in Section 5.2 and calculate their risk level based on the set of sufficient attacks to realize them. First we conduct a risk assessment without any countermeasures considered in Section 7.2 and then in Section 7.4 we recalculate the severity and likelihood of attacks when the proposed countermeasures from Chapter 6 are applied, and finally determine the risk levels with countermeasures.

7.2 Attacking a swarm member vs. attacking the swarm

The application of swarms of smaller robots, with limited capabilities comes from the idea that these robust and failure tolerant systems can be used in safety-critical missions, where the failure of a fraction of the swarm members does not have a severe impact on the whole swarm's mission. However, when the size of the swarm is not measured in tens or hundreds of robots, even attacking a single member can affect the mission.

Having a limited number of swarm members does not mean that the mission will be less efficient, since their behaviour is optimized according to the size of the swarm, and in lots of cases when there is a scarcity of operational space, it is better to use smaller swarms. In the case of swarm with a handful of members, the trade-off is endangering the success of the mission by one or several members' failure or misbehaviour, either caused by adversaries or other, natural causes like hardware faults, software bugs or environmental conditions.

The consequences of attacking a single member of the swarm could include

- Reduced efficiency in executing the mission – since the behaviour, distribution, etc. is optimized for a fixed number of members;
- Physical harm to other swarm members due to collisions;
- Change in the expected swarm behaviour - in some configurations swarm members could trigger events which could result in switching to other behaviours;
- When there is a clearly established hierarchy among swarm members, attacking the master members could severely abuse the mission.

The last point brings us to the second option, to perform an attack against the whole swarm. Hijacking the leading member (if present) is on the border of these categories since it could result in an attack against the whole swarm. Of course, attacking the whole swarm is a very clear way to stop its mission or delay it for a sufficiently long time. However, it may require less effort to just disable one or few members to mislead or completely disrupt the swarm.

If an attack against one swarm member, once successful, can be performed again and again against the rest of the swarm members with little additional effort, the severity of the attack increases significantly. Likelihoods are not affected.

7.3 Risk assessment

First, for each low-level attack (presented in Section 5.3) we determine their likelihood and severity, so we can calculate the risk related to them as seen in Table 5 below.

ID	Attack	Likelihood	Severity	Risk Level
A1	Brute force credentials	Unlikely	2	Low
A2	Exploit weakness in protocol	Likely	2	Medium
A3	Jam communication channel	Likely	2	Medium
A4	Spoof authorized stop	Likely	3	High
A5	Exploit weakness in platform	Likely	2	Medium
A6	Sign firmware image	Unlikely	2	Low
A7	Impersonate target	Likely	1	Low
A8	Predict path	Likely	2	Medium
A9	Exhaust consumables	Likely	2	Medium
A10	Spoof sensor data	Likely	2	Medium
A11	Build valid firmware image	Unlikely	3	Medium
A12	Gain physical access	Unlikely	3	Medium
A13	Incapacitate swarm member	Likely	2	Medium
A14	Modify environmental conditions	Likely	2	Medium
A15	Steal sensitive equipment	Unlikely	3	Medium
A16	Trigger component failure	Unlikely	2	Low
A17	Mislead operator	Unlikely	1	Low
A18	Install malware on operator environment	Likely	1	Low
A19	Steal operator credentials	Likely	2	Medium
A20	Exploit weakness in software	Likely	2	Medium

Table 5 Risk assessment of attacks

From the underlying attacks, we can estimate the overall risk level of the threat scenarios by choosing the highest risk available from the risks identified for basic attacks above.

Attacker goal	Related attacks	Overall risk
Sabotage mission	A1, A2, A3, A4, A5, A6, A9, A11, A12, A14, A15, A16, A17, A18, A19, A20	High
Cause financial damage to swarm operator	A1, A2, A3, A4, A5, A6, A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18, A19, A20	High
Cause physical harm to human being or property	A1, A2, A4, A5, A6, A10, A11, A14, A15, A16, A17, A18, A19 A20	High
Disturb environment or bystanders	A1, A2, A4, A5, A6, A10, A11, A14, A15, A16, A17, A18, A19 A20	High
Steal swarm member	A2, A4, A5, A6, A10, A11, A14, A15, A16, A17 A18, A19, A20	High
Steal sensitive data	A1, A2, A5, A6, A11, A15, A18, A19, A20	Medium

Table 6 Attacker goals, related attacks and their overall risk

As we can see, five from the total six attacker goals we identified have a high risk, since the attack which has a high risk (A4) can help realize them. In the next section we redo the risk assessment by taking the proposed countermeasures into account.

7.4 Risk assessment with countermeasures

First we look at whether a low-level attack can be mitigated and if so, with what type of countermeasure from the ones described in Chapter 6. Then we recalculate the likelihoods and hence the risk level of these attacks. Results are presented in Table 7.

ID	Attack	Countermeasure	Likelihood	Severity	Risk Level
A1	Brute force credentials	Communication	Unlikely	2	Low
A2	Exploit weakness in protocol	Communication	Unlikely	2	Low
A3	Jam communication channel	Communication	Unlikely	2	Low
A4	Spoof authorized stop	Communication	Unlikely	3	Medium
A5	Exploit weakness in platform	Hardening	Unlikely	2	Low
A6	Sign firmware image	Deployment	Unlikely	2	Low
A7	Impersonate target	-	Likely	1	Low
A8	Predict path	-	Likely	2	Medium
A9	Exhaust consumables	-	Likely	2	Medium
A10	Spoof sensor data	-	Likely	2	Medium

A11	Build valid firmware image	-	Unlikely	3	Medium
A12	Gain physical access	-	Unlikely	3	Medium
A13	Incapacitate swarm member	-	Likely	2	Medium
A14	Modify environmental conditions	-	Likely	2	Medium
A15	Steal sensitive equipment	-	Unlikely	3	Medium
A16	Trigger component failure	-	Unlikely	2	Low
A17	Mislead operator	-	Unlikely	1	Low
A18	Install malware on operator environment	-	Likely	1	Low
A19	Steal operator credentials	-	Likely	2	Medium
A20	Exploit weakness in software	-	Likely	2	Medium

Table 7 Attacks with recalculated risks after applying countermeasures

Please note that some new countermeasures will be introduced against use-case specific attacks in D4.8.

Now, we recalculate the overall risks for the attacker goals in Table 8.

Attacker goal	Related attacks	Overall risk
Sabotage mission	A1, A2, A3, A4, A5, A6, A9, A11, A12, A14, A15, A16, A17, A18, A19, A20	Medium
Cause financial damage to swarm operator	A1, A2, A3, A4, A5, A6, A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18, A19, A20	Medium
Cause physical harm to human being or property	A1, A2, A4, A5, A6, A10, A11, A14, A15, A16, A17, A18, A19 A20	Medium
Disturb environment or bystanders	A1, A2, A4, A5, A6, A10, A11, A14, A15, A16, A17, A18, A19 A20	Medium
Steal swarm member	A2, A4, A5, A6, A10, A11, A14, A15, A16, A17 A18, A19, A20	Medium
Steal sensitive data	A1, A2, A5, A6, A11, A15, A18, A19, A20	Medium

Table 8 Recalculated risk levels to attacker goals

As we can see, all risks are now reduced to medium as a consequence of applying countermeasures and thus eliminating high risks.

8 Future work

This chapter describes future work that will be refined in the next and final third of the project that will be presented in the final version of this deliverable, D4.8 - Final security threat and attack models, where our focus will be shifted to use-case specific threat analysis and risk assessment.

8.1 Safety

There has already been much work included concerning safety in this deliverable – see Section 3.1.2 on industry standards on safety; assets and attacks concerning human and environmental safety in Chapters 4 and 5; and some safety related countermeasures already included in Chapter 0. Since there will be a Safety workshop organized by the consortium not so long after the submission of this deliverable – and threats concerning safety and their corresponding countermeasures are heavily dependent on the application field of the swarm, we chose to dedicate a standalone chapter to safety in D4.8.

8.2 Applicability of assets on designated project use cases

In Chapter 4 we have described assets to a generic mission which involves swarm(s) of CPSs and various CPSwarm Workbench Tools and runtime environment components. These will be further extended, refined and matched to the three use-case scenarios of this project. It will be followed by analyses of threats specific to the three different application scenarios. New attack trees will be created and the old ones will be matched to each of the scenarios. There will be separate risk assessments defined for each scenario, based on Chapter 0 from this deliverable, but the likelihood and estimated severity of risks will be matched to the corresponding application scenario.

8.3 Evaluation guidelines

For commercial deployment of swarms in real, production environments a high level of security assurance is required. While no legal framework exists as of now for most application areas that govern the security requirements for CPS, work is underway both in EU organizations and worldwide on developing a common set of requirements and criteria that can be used when evaluating IoT devices. As a contribution to this effort and as an extension of the existing methodology used by SLAB (see Meforma [17]) and as part of an ongoing effort to develop a commercial certification scheme that can be used to describe the security level of these devices, we plan to develop an evaluation profile for drones and other autonomous robots. This profile and its requirements will be described in the next version of the deliverable, and will connect with work done in parallel as part of the VESSEDIA [18] project.

8.4 Attack tree modelling in Modelio

As was previously mentioned in Chapter 5.1, we plan to develop a module for Modelio to help users of the workbench perform security modelling within the same ecosystem. This not only makes it easier for engineers working with the workbench to systematically describe threats, the mere presence of the feature prompts users to explore their own threat landscape and include the results of their findings in their development work.

9 Bibliography

[1] [Online]. Available: https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx.

[2] [Online]. Available: https://en.wikipedia.org/wiki/DREAD_(risk_assessment_model).

[3] [Online]. Available: http://www.octotrike.org/.

[4] [Online]. Available: https://www.safaribooksonline.com/library/view/risk-centric-threat/9780470500965/c06.xhtml.

[5] Agarwal et al., «VAST Methodology: Visual, Agile, and Simple Threat Modeling.,» 2016.

[6] «The COSSIM H2020 project,» [Online]. Available: http://www.cossim.org.

[7] C. Schmittner et al., «Security Application of Failure Mode and Effect Analysis».

[8] Dr. J. Marshall, «An Introduction to Fault Tree Analysis».

[9] [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:31985L0374&from=EN.

[10] [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32001L0095&from=EN.

[11] «The Robolaw project,» [Online]. Available: http://www.robolaw.eu/publicdocs.htm.

[12] [Online]. Available: http://www.beuc.eu/publications/beuc-x-2017-039_csc_review_of_product_liability_rules.pdf.

[13] [Online]. Available: https://ec.europa.eu/transport/sites/transport/files/com20160766_en.pdf.

[14] [Online]. Available: http://www.reersafety.com/pt/en/safety-guide/safety-in-the-working-environment/iec-62061-sil.

[15] «Attack Defense Tree tool,» [Online]. Available: http://satoss.uni.lu/members/piotr/adtool/.

[16] «The TREsPASS FP7 project,» [Online]. Available: https://www.trespass-project.eu/.

[17] E. Jeges, B. Berkes, B. Kiss and G. Eberhardt, «MEFORMA Security Evaluation Methodology,» 2014.

[18] «The VESSEDIA H2020 project,» [Online]. Available: https://vessedia.eu.