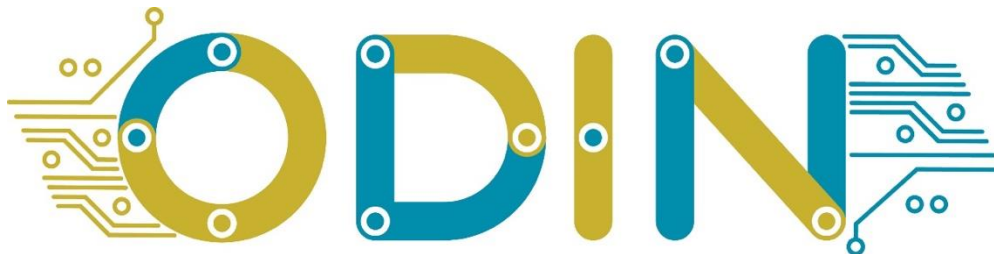


Open-Digital-Industrial and Networking pilot lines using modular components for scalable production

Grant Agreement No : 101017141
Project Acronym : ODIN
Project Start Date : 1st January, 2021
Consortium : UNIVERSITY OF PATRAS – LABORATORY FOR MANUFACTURING SYSTEMS AND AUTOMATION
FUNDACION TECNALIA RESEARCH & INNOVATION
KUNGLIGA TEKNISKA HOEGSKOLAN
TAMPEREEN KORKEAKOULUSAATIO SR
COMAU SPA
PILZ INDUSTRIE ELEKTRONIK S. L.
ROBOCEPTION GMBH
VISUAL COMPONENTS OY
INTRASOFT INTERNATIONAL SA
GRUPO S21SEC GESTIÓN, S.A.
FUNDACION AIC AUTOMOTIVE INTELLIGENCE CENTER FUNDAZIOA
DGH ROBOTICA, AUTOMATIZACION Y MANTENIMIENTO INDUSTRIAL SA
PSA AUTOMOBILES S.A.
AEROTECNIC COMPOSITES SL. U.
WHIRLPOOL EMEA SPA



Title : ODIN Networked Component initial prototype
Reference : D4.1
Availability : Public
Date : 30/06/2022
Author/s : S21SEC, INTRA
Circulation : EU, Consortium

Summary:

The purpose of this document is to present the design and initial prototype of:
a) OpenFlow communication and integration architecture and
b) Active DT Protection Framework and DT Intelligent Threat Analysis Toolkit

Table of Contents

LIST OF FIGURES	4
LIST OF TABLES	5
EXECUTIVE SUMMARY	6
1. INTRODUCTION	8
2. OPENFLOW	10
2.1. Introduction.....	10
2.2. Features.....	11
2.2.1. Orchestrate Modules and Resources	11
2.2.2. Emulation.....	12
2.2.3. Simulation	12
2.2.4. React on Shopfloor Events.....	13
2.2.5. React on Safety Events	13
2.2.6. React on Security Events	13
2.2.7. Control & Monitor Task and Action Execution Flow	14
2.2.8. Monitor Network Software Modules Status	15
2.2.9. Control OpenFlow Execution Flow	15
2.2.10. Request Replanning	16
2.2.11. Validate Open Schedules	16
2.2.12. OpenFlow Knowledge Repository.....	17
2.2.13. Information Exchange with ERP systems.....	18
2.2.14. User Interface.....	19
2.3. Design	26
2.4. Initial Prototype	27
3. CYBERSECURITY.....	29
3.1. Introduction.....	29
3.2. Features.....	29
3.2.1. Threat modelling and attack surface definition	29
3.2.2. Detection	30
3.2.3. Response	30
3.3. Design	31
3.3.1. Attack modelling methodology.....	31
3.3.2. ODIN network architecture.....	31

3.3.3. Cybersecurity Module Architecture.....	32
3.3.4. Validation.....	34
3.4. Initial Prototype	34
3.4.1. Prototype environment.....	34
3.4.2. Attack scenario.....	35
3.4.3. Cybersecurity module implementation	41
3.4.4. Initial integration and testing	43
4. CONCLUSIONS	48
5. GLOSSARY	49
6. REFERENCES	50
7. ANNEX A: MAGMA ADAPTATION TO INDUSTRIAL NETWORKS	51
7.1. Steps of the Analysis.....	51
7.2. Step 1 - Definition of the Scope.....	53
7.3. Step 2 - Identification of the relevant assets and Entry Points	55
7.3.1. Central Control Station	55
7.3.2. Certification Authority	55
7.3.3. Historian Database	56
7.3.4. Control Station	56
7.3.5. Controller	56
7.4. step 3 - Analyse Potential Drivers and References in MaGMa	58
7.4.1. Potential Drivers and References in the Ros-based Scenario	58
7.5. Step 4 – Generation of the L1 Use Cases	59
7.5.1. L1 Use Cases for the ROS-based Scenario.....	60
7.6. Step 5 – Generation of the L2 Use Cases	60
7.7. L2 Use Cases for the ROS-based Scenario.....	62
7.8. Step 6 – Generation of the L3 Use Cases	64
7.8.1. L3 Use Cases for the ROS-based Scenario.....	64
7.9. Interpretation of the Metrics	66
7.10. References.....	68

LIST OF FIGURES

Figure 1: ODIN Reference Architecture - Component Level Diagram.....	9
Figure 2: OpenFlow modules orchestration.....	11
Figure 3: OpenFlow: Request Replanning.....	16
Figure 4: OpenFlow KR: User, Product Plan, Schedule Data Model.....	17
Figure 5: Information Exchange with AEROTECNIC - SAP.....	19
Figure 6: OpenFlow UI: Login Page.....	19
Figure 7: OpenFlow UI: Schedules Tab.....	20
Figure 8: Open Flow UI: Execution Status Tab.....	20
Figure 9: Options while Schedule is running.....	21
Figure 10: Options while Schedule is paused.....	21
Figure 11: Tasks Execution Status.....	21
Figure 12: Actions Execution Status.....	22
Figure 13: White Goods preliminary pilot case demo - Tasks diagram.....	23
Figure 14: Open Flow UI: Product Plans Tab.....	24
Figure 15: Open Flow UI: Resources Tab.....	24
Figure 16: Open Flow UI: Resource's modules.....	25
Figure 17 : OpenFlow Initial Prototype Interfaces Design.....	26
Figure 18: ODIN Preliminary White Goods Pilot Case deployment diagram.....	27
Figure 19: ODIN Network Architecture based on IEC 62443 / ISA99 model.....	32
Figure 20: ODIN Network Architecture with Cybersecurity tools integrated.....	33
Figure 21: ODIN Networked Component Emulation.....	35
Figure 22: ODIN Architecture interactions flow.....	36
Figure 23: OpenFlow publishers and subscribers list.....	37
Figure 24: MITRE Matrix filtered for ICS domain.....	39
Figure 25: ODIN Cybersecurity System general implementation.....	41
Figure 26: ODIN Cybersecurity System detailed implementation.....	43
Figure 27: ODIN Cybersecurity System detailed implementation.....	44
Figure 28: Event detected in the SIEM.....	44
Figure 29: Automated case management in Shuffle.....	45
Figure 30: Alert management in The Hive.....	46
Figure 31: Alert scalation to case in The Hive.....	47
Figure 32: Steps of MaGMA use case framework for a given scenario.....	52
Figure 33: Relative reference in the formulas of the MaGMA tool.....	53
Figure 34: Modified formulas in the MaGMA tool.....	53
Figure 35: Use case architecture diagram.....	54
Figure 36: Threat categories proposed in MaGMA to serve as an overview of the use cases.....	59
Figure 37: Examples of the L2 use cases provided by MaGMA.....	61
Figure 38: Example of the selected L2 use cases for the ROS-based scenario.....	63
Figure 39: Example of the generated L3 use case for the ROS-based scenario.....	65
Figure 40: Colour scale used in MaGMA.....	66
Figure 41: Example of an optimal deployed use case.....	67
Figure 42: Generated dataset of the metrics defined in MaGMA.....	67

LIST OF TABLES

Table 1: OpenFlow features	10
Table 2: White Goods preliminary demo, ActionLib server ROS Nodes	12
Table 3: OpenFlow Simulating Interfaces	13
Table 4: OpenFlow security interface	14
Table 5: OpenFlow Implemented Action Interfaces.....	14
Table 6: White Goods preliminary demo tasks.....	18
Table 7: OpenFlow UI: Navigation Tabs.....	19
Table 8: Summary of the assets in the network of the use case and their properties.....	57

EXECUTIVE SUMMARY

This deliverable is the result for M18 initial prototype of the ODIN Network Component, comprising the outcome of tasks T4.1 “Reference integration and communication architecture for reconfigurable production” and T4.2 “Cybersecurity and data processing in autonomous production environments”. The document describes the “ODIN Networked Component” including a) OpenFlow module, responsible to integrate, orchestrate, manage and coordinate production resources to execute manufacturing schedules, and b) the Cybersecurity module responsible to provide detection and response capabilities on the deployed Network Component.

The initial prototype of the OpenFlow module is a functional initial prototype able to integrate, orchestrate and manage other modules. The initial prototype of OpenFlow module includes the following functionalities:

- Orchestration of Modules and Resources,
- Emulated execution of a production schedule,
- Simulated execution of a production plan in a 3D virtual environment,
- Reaction on Shopfloor Events (Execution failure events and recovery strategies),
- Reaction on Safety Events (Safety violation events and recovery strategies),
- Reaction on Security Events: (Security violation events and recovery strategies),
- Control, Monitor Task and Action Execution Flow,
- Monitoring of Network Software Modules Status,
- Controlling of OpenFlow Execution Flow,
- Request execution task replanning,
- Validation of Open Schedules,
- OpenFlow Knowledge Repository,
- Information Exchange with ERP systems,
- UI offering control, monitoring, and views of OpenFlow functionalities to end user.

A prototype for ODIN Cybersecurity solution is also described and including the process and methodology for ODIN threat modelling and describing a cybersecurity toolkit for incident detection and response. In particular, the modelling and monitoring protection will be focused on the scope of the ODIN Networked component (OpenFlow).

Different methodologies for threat modelling, like MaGMA [15] and MITRE ATT&CK [13] have been analyzed, and using both approaches it has concluded on a particular attack modelling methodology definition. The resultant threat model for ODIN will include a set of selected techniques that can be used for a hypothetical attack to the ODIN Networked component.

Two main components of the Cybersecurity module are. Incident detection and Incident Response. In terms of detection, the implementation of detection is based on Security Information and Event Management (SIEM) tools, with the capabilities of collection of raw data from the network and systems and event generation. In terms of response, the implementation is based on Security Orchestration, Automation and Response (SOAR) tools, with the capabilities of automation workflows definitions that derive in incident response and further Security Operation Center SOC management approach.

A proposal for the ODIN network architecture, coherent and based on former ISA 99, and recent IEC 62443 and NIST 800-82 as reference frameworks, that proposes security in the

design of industrial networks, has been also including highlighting the SIEM and SOAR components on the deployed architecture.

Initial attack scenario has been defined along with the attack surface analysis to identify the elements with which the OpenFlow interacts, and the interfaces through which an attack exploit could be performed. An adapted threat model is designed, using Cyber Kill Chain for OpenFlow, and different applicable tactics are selected from the MITRE ATT&CK and MaGMA frameworks, included as an ANNEX.

Cybersecurity solution is described as deployed and integrated in the prototype environment. Actual stage of the project includes a SIEM agent in an endpoint with the emulated OpenFlow, that collects logs, detects the security events and sends them to the SIEM server, where this information is collected, normalized and correlated, so that security alerts are raised based on their criticality. These alerts are latter sent to the SOAR system, where they are further investigated to allow the appropriate case management and reactive response.

SIEM and SOAR interconnection are built up using opensource software with Wazuh and TheHive, described as cybersecurity toolkit. Finally, an example of the Brute Force technique is presented, showing the full chain of detection and response by the ODIN Cybersecurity System for this attack.

1. INTRODUCTION

This deliverable describes the concept, features design and implementation of the ODIN Network Component, focusing in the initial prototype as well as presenting the design of the final version. The design and development of the Network Component follows an agile approach and its design follows closely the design and development of the other ODIN modules as well as the ODIN Pilot Cases. The ODIN Network Component is presented from the perspective of its two modules. More precisely the OpenFlow initial prototype is presented in section 2 and the Cybersecurity module that is presented in section 3.

The OpenFlow initial prototype is comprised of the initial prototypes of the Knowledge Repository, the OpenFlow Core component, the OpenFlow emulation engine and the OpenFlow UI. It is responsible to integrate the software system of the ODIN architecture and orchestrate ODIN modules to robustly and efficiently execute the production schedule.

The OpenFlow is a functional initial prototype that also demonstrates integrated functionalities. In particular it has been used in the preliminary White Goods small case demo (M12) and in M18 White Goods and Automotive small scale integrated demos in M18 of project development. The initial prototype used as input the ODIN Reference architecture described in D1.4 and provides most of the presented functionality in Network Component of D1.4 too. The functionalities which have been used in the M18 Pilot Cases are presented in Section 2.

ODIN OpenFlow is a robust integration platform that orchestrates and monitors Human-Robot Collaboration (HRC) systems and their modules to safely execute a manufacturing process and respond to real-time unprecedented events taking place through the process. The ODIN OpenFlow based integrated system is composed by four main components. Figure 1 presents the high-level abstract components described in ODIN Reference Architecture in D1.4.

The transition of ODIN features from WP2 and WP3 to actual data models and interfaces is based on technological partners' collaboration. These interfaces were described in ODIN Project Architecture and are presented in Section 2.2.7.

The initial prototype of OpenFlow has been released as a docker image hosted in a private docker repository of LMS for distribution only inside the ODIN consortium and for the project needs. Further to this version, an initial version was developed for the preliminary White Goods demo that allowed integration and testing. Additionally, it was used to get early feedback from developers and pilot case responsible partners.

More information is provided in section 2 of this document, which provides a detailed overview of the current implementation state of OpenFlow module as well as the direction for future development, although the design and planning of the developments in WP4 mostly take place in an agile way. In particular Section 2 describes the architecture followed in the implementation of OpenFlow and presents the following:

- Design and development of OpenFlow sub-modules.
- Implemented interfaces to described server modules presented in ODIN Project Architecture.
- Distribution of ODIN OpenFlow integrated system through docker images.
- ODIN implemented features offered by the current implementation of OpenFlow module.

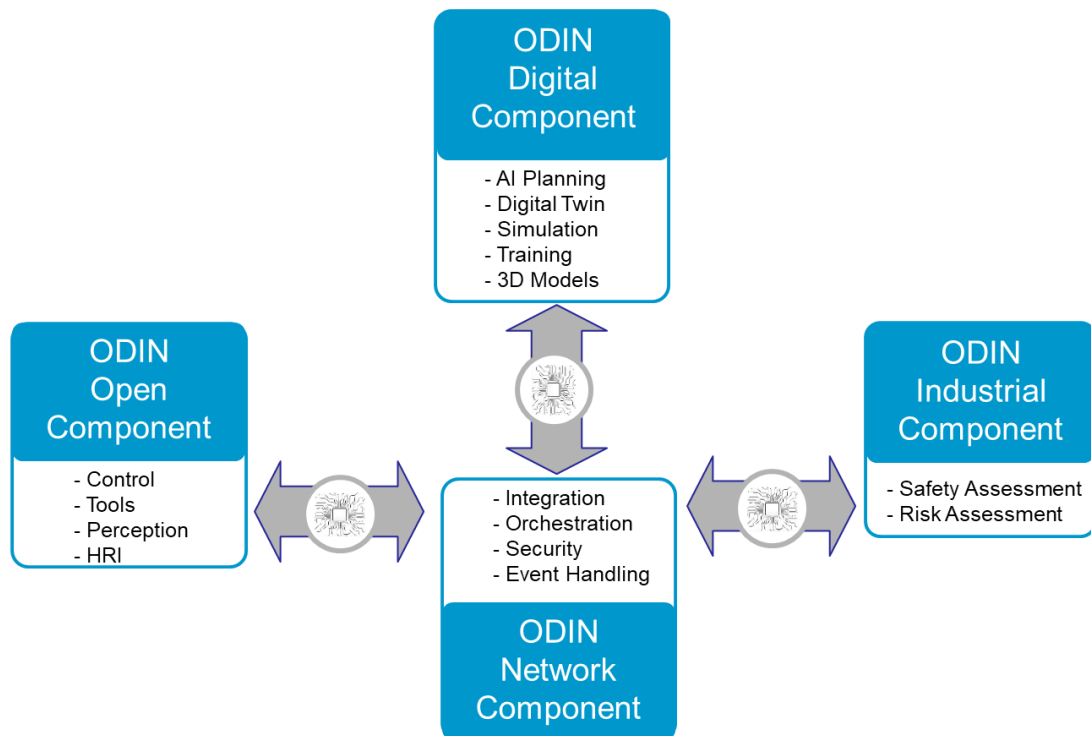


Figure 1: ODIN Reference Architecture - Component Level Diagram

The ODIN Cybersecurity initial prototype is described in section 3. Section 3.2 presents its three main features, namely the threat modelling, detection and protection. Section 3.3 presents the design of the system, through the attack modelling methodology, the solution architecture and the validation. Finally, section 3.4 describes the initial prototype through the prototype environment, attack scenario and detection and response solution.

2. OPENFLOW

2.1. Introduction

This section aims to present the OpenFlow first prototype module, and details of the orchestration process that takes place under the hood. In addition, this section describes the user interface (UI) of OpenFlow first prototype, including the visualization of the orchestration process. The first prototype version of the OpenFlow module has been based on the OpenFlow Architecture as well as the ODIN Architecture specifications of D1.4. OpenFlow interoperates and manages different ODIN modules to the features that are presented in Section 2.2. These features were presented in D1.4 and are implemented through the development phase of WP4. Table 1 summarizes the OpenFlow features.

Table 1: OpenFlow features

<u>OpenFlow Features</u>
Orchestrate Modules and Resources
Emulation
Simulation
React on Shopfloor Events
React on Safety Events
React on Security Events
Control & Monitor Task and Action Execution Flow
Monitor Network Software Modules Status
Control OpenFlow Execution Flow
Request Replanning
Validate Open Schedules
OpenFlow Knowledge Repository
Information Exchange with ERP systems
User Interface

Due to its modular architecture, the OpenFlow integration software system is flexible and extensible to support, with low effort, new functionalities and adjustments to modules that aroused through the development and testing phase in small-scale pilot cases. This also adheres to the norm induced from the increasing product variety in an industrial environment [3] and the need of mass customization to be able to handle such ranges in different products manufacturing process [1,2].

Following the context of modern Industry 4.0, OpenFlow modules offer interfaces for the actuation of Actions and their “actuator” subject. For instance, OpenFlow can orchestrate actuator submodules by communicating with them and initiating an actuation model by an Action or cancel an already ongoing action. The design and integration principles for such behaviour are presented in Section 1.1.

Finally, Section 2.4 presents the current version of OpenFlow, with the designed and implemented features through WP4 until the scope of this deliverable on M18 of the project.

2.2. Features

This section describes the currently designed features as well as the implementation status of the first OpenFlow prototype.

2.2.1. Orchestrate Modules and Resources

Orchestrating different modules is an essential process in a human-robot collaborative environment, and it has been demonstrated in use cases derived from the automotive industry in which human operators involved into and were part of the manufacturing process [6,7].

OpenFlow is the module responsible for the orchestration of other modules, as described in the ODIN Reference Architecture in D1.4. An OpenFlow based system is composed from different submodules that have to be monitored and orchestrated. OpenFlow utilizes peer-to-peer and publish-subscribe communication protocols to enable centralized control of its submodules. The orchestration engine that manages OpenFlow modules is part of the OpenFlow Core submodule, and it is responsible for the features presented in this section. The main objective of the orchestration module is to successfully execute a production Schedule in order to complete a pre-defined manufacturing process. Therefore, besides orchestration, OpenFlow takes into account real-time data about the availability and suitability of manufacturing resources, to safely execute the required actions.

The OpenFlow is capable to change the execution flow of a production process, react to events and dispatch appropriate actions to ensure that the production is complete.

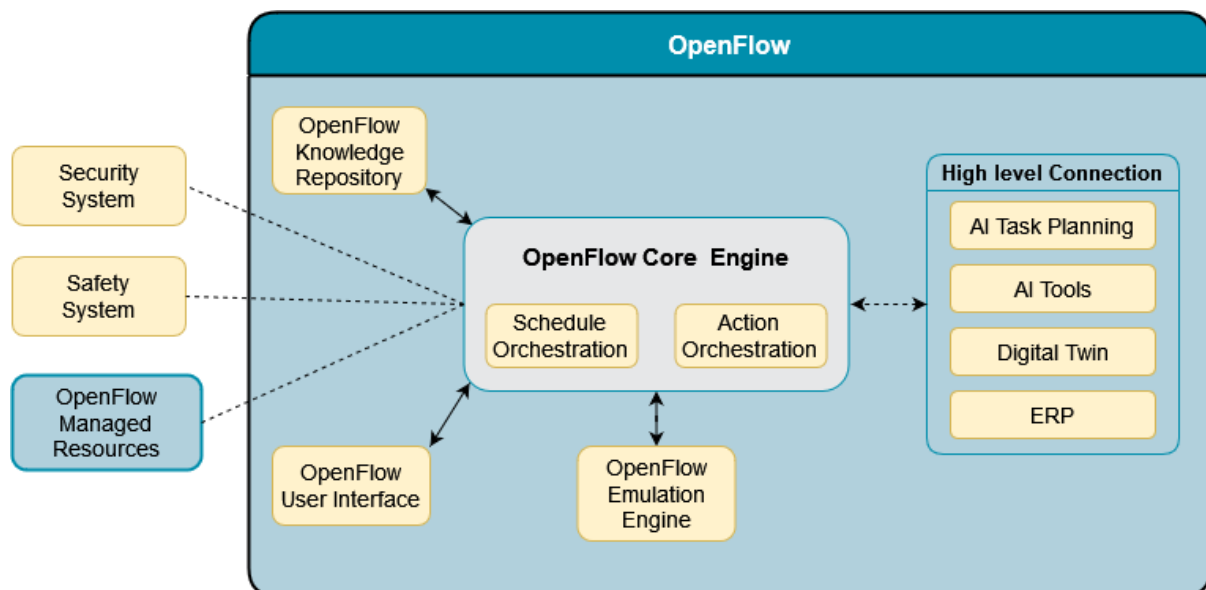


Figure 2: OpenFlow modules orchestration

Figure 2, depicts in high level the internal modules of OpenFlow. The OpenFlow User Interface is described in detail in 2.2.14. The OpenFlow Knowledge Repository, which is responsible for persisting the required data is presented thoroughly in 2.2.12. OpenFlow Emulation Engine can imitate the execution of a production Schedule in an emulated environment for testing purposes by emulating all necessary interfaces and resources and it is presented in section 2.2.2.

Planning the production is a key functionality of production system. OpenFlow integrates and interoperates closely with the AI Task Planner module which is in charge of planning and

replanning the Tasks that need to be performed. For the purpose of execution, these Tasks into Actions are converted in executable production Schedules. This process is depicted in 2.2.10, while the Task Planner User Interface allows the User to manually adjust and create a production Schedule.

2.2.2. Emulation

The OpenFlow includes an emulation engine that can create and start responsive emulated interfaces for all OpenFlow managed resources (Figure 2). The OpenFlow emulation engine emulates the interfaces of other modules and their responses providing a realistic emulation environment with an API identical to the real one.

In this way the OpenFlow can setup an emulated (virtual) environment that is comprised of emulated modules and execute an OpenFlow production Schedule in the emulated environment.

The emulation is focused in the communication and information exchange between modules. Execution of Schedules in virtual environments where the modules are represented in more detail, including the space geometry can be handled also by OpenFlow in the simulation functionality that is covered in section 2.2.3.

In addition to the actual process orchestration in real production applications, the need of emulating a robotic environment is also important in multiple occasions. Emulation allows for validation of the cohesion between Schedules Actions prior to their actual execution. The emulation can run not only in the nominal duration of the real time execution but also in much less time. This means that an emulated execution of a complete schedule only requires a fragment of the time the actual execution takes, thus speeding up the testing and development process.

The emulated modules comply with ROS and have specifically assigned node names.

For instance, the emulated module interfaces that are managed and emulated by OpenFlow for the successful implementation of the White Goods demo in an emulating environment are displayed in Table 2 below.

Table 2: White Goods preliminary demo, ActionLib server ROS Nodes

Action Name	ActionLib server ROS Node path
Execute Human Task	/emulation/operator_support/integration/node/execute_human_task
Control Gripper	/emulation/gripper/integration/node/control_gripper
Control Tool changer	/emulation/tool_changer/integration/node/control_toolchanger
Move Arm Joint	/emulation/cobot/integration/node/move_arm_joint
Configure Payload	/emulation/configure_payload/integration/node/configure_payload
Configure TCP	/emulation/configure_tcp/integration/node/referenced_execution

2.2.3. Simulation

While Emulation (presented in section 2.2.2) is mimicking the complete sequence of steps required for the execution of a production Schedule, which is beneficial for testing and development phases, it is essential to further test a production Schedule on a simulated environment prior to its actual deployment in a real scale robotic application. A simulated environment visualizes the real environment into a 3D software world in which the actual movement of a robot resource can be tracked, and its interference can be tested with any other physical 3D object in its surroundings. Simulation's goal is validating and verifying any

concerning factors in a robotic product line environment prior to their actual installation so that the optimal configuration can be selected [2].

OpenFlow can connect and control resources that are simulated. The actions required for the completion of a production Schedule on a real or emulated environment, are interfaced in a simulating mode too. Additionally, specific simulating actions have been designed and implemented to interface and aid the management of 3D objects. Table 3 below presents the interfaced Actions that initiate a simulation and spawn or remove an object in the simulation scene.

Table 3: OpenFlow Simulating Interfaces

Client Module	Implemented Interface	Description	Server Module
OpenFlow	Simulate	Initiates simulation	Digital Simulation
OpenFlow	Spawn Dynamic Object	Spawns a Dynamic Object on a simulation scene at specific Pose.	
OpenFlow	Vanish Dynamic Object	Removes Dynamic Object from a scene.	
OpenFlow	Control Assembly Hierarchy	Control Assembly Hierarchy in Simulation Environment	

2.2.4. React on Shopfloor Events

Validated use-cases have shown that monitoring of shopfloor events is a highly regarded factor to consider while designing and implementing mobile robotic applications in the automotive industry [4].

OpenFlow module is capable of coordinating and orchestrating external resources to monitor and respond Shopfloor Events. In current iteration of D4.1 through M18, a first prototype with basic functionality has been implemented and will be further developed and integrated with the appropriate models to react on shopfloor events in the future.

2.2.5. React on Safety Events

OpenFlow has a dedicated Safety module to address safety Events that can arise at any stage during the execution of a production Schedule. This module has a very short latency to capture and address security triggers as fast as possible and provide solutions to recover the system and continue the manufacturing production. In current development stage, there is a test safety module which will be further improved and developed to address actual safety events in the future.

2.2.6. React on Security Events

In a production line environment in which human and robots share workspace [5], the need to react on security induced events is essential.

OpenFlow offers interfaces to specific event topic listeners and can monitor and evaluate arrived message and provide specific tailored response to address such events. Responses to events may include the following:

- Email notifications about security events.
- Actuation of other implemented action interfaces required to address a security raised event.

- Stop or resume a production Schedule according to the security event type or severity.
- Notify Operators about security events.

To address the Security events OpenFlow implements the following interface for the security topic as described in ODIN Project Architecture.

Table 4: OpenFlow security interface

Client Module	Implemented Topic Interface	Server Module
OpenFlow	Security Event	Cyber Security

Messages that arrive to this topic include specific event ID, event type and severity level upon which a Subscriber ROS Node on OpenFlow will associate specific patterns and raise the respective alert Events.

2.2.7. Control & Monitor Task and Action Execution Flow

OpenFlow has the ability to coordinate, monitor and execute a stored production Schedule through its orchestrator module. Orchestrator leverages the Actor model to handle the actuation of Schedule's actions as this offers immutable data exchanging during execution which is considered a quite robust feature for communication in a robotic environment [8].

OpenFlow utilizes customizable data models for each implemented action. Such data models use the required interface protocol to connect with different modules interfaces. In ROS-based systems, such as the ODIN, the ActionLib protocol is often used because it allows Clients to control, configure action's goal and receive status updates during and after the execution of each action [8]. Multiple interfaces are currently consumed by the OpenFlow module. Table 5 below contains the interfaces whose clients are currently offered by OpenFlow to the modules described in ODIN Project Architecture.

Table 5: OpenFlow Implemented Action Interfaces

Client Module	Implemented Interface	Server Module
OpenFlow	Move Arm to TF Frame	Cobot
OpenFlow	Move Arm Joint	
OpenFlow	Move Arm Cartesian	
OpenFlow	Control Arm Mode	
OpenFlow	Execute Skill	Easy Programming
OpenFlow	Execute Skill Referenced	
OpenFlow	Configure Tools	End Effector
OpenFlow	Control Arbitrary Tool	
OpenFlow	Control Gripper	
OpenFlow	Control Tool Changer	
OpenFlow	Trigger Screwdriver	
OpenFlow	Detect Object	Environment Perception
OpenFlow	Configure Detection	Human Detection
OpenFlow	Move Arm Cartesian	Mobile Robot
OpenFlow	Move Arm Joint	

Client Module	Implemented Interface	Server Module
OpenFlow	Navigate	
OpenFlow	Control Trajectory Tracking	Operator Support
OpenFlow	Execute Task Synchronous	
OpenFlow	Execute Task Synchronous Referenced	
OpenFlow	Operator Support display configuration	
OpenFlow	Show Notification	
OpenFlow	Set Safety Border Projection	
OpenFlow	Set Light Indication Projection	
OpenFlow	Set Preset UI Projection	
OpenFlow	Set Instructions Projection	
OpenFlow	Set Virtual Buttons Projection	
OpenFlow	Unset Any Projection	
OpenFlow	Cartesian Goal Motion Control	Task Planning
OpenFlow	Joint Goal Motion Control	
OpenFlow	Referenced Goal Motion Control	
OpenFlow	Task Planning	

The actions Clients are implemented with extensibility and maintainability in mind so that any new requirements (e.g., new field attributes in actions definition) can be easily integrated into the existing data models with minimal effort. Additionally, configuration options, such as if an action can be paused or not while its active and what actions may be executed after each action, offer control upon the Schedule's execution flow. These configuration options are persisted alongside the Schedule's actions in Knowledge Repository 2.2.12.

OpenFlow UI takes over of OpenFlow's under the hood features and offers a user-friendly interface to visualize execution flow of a Schedule to user by displaying status for each Task and Action of a production Schedule. OpenFlow UI is thoroughly presented in 2.2.14.

2.2.8. Monitor Network Software Modules Status

As OpenFlow system is itself a modular system and part of the ODIN software system, it can constantly monitor the status of other modules and communicate with specific actuator components of other modules to address temporary failures. This feature is currently under development and will be included in feature releases. The current draft implemented version can react when a module's software interface cannot be accessed by attempting to reconnect.

This behavior is configurable and can be customized to fit different scenarios.

Future versions will be able to detect in advance the status of specific modules and the available services and react on specific occasions. For instance, trigger a rescheduling in case a cobot module is not available so that the AI Task Planner could request a human to execute a task instead.

2.2.9. Control OpenFlow Execution Flow

OpenFlow Core has the ability to initiate the execution of a Schedule or pause, stop or cancel it while its running and resume gracefully when it is paused. To achieve such functionality OpenFlow implements interfaces for the following ROS Action Servers:

- Start New OpenFlow Schedule Execution

This service takes as input the id of a stored Schedule inside Knowledge Repository or a completely new Schedule in json format and starts its execution, while it allows for execution status to be updated through feedback and result definition in Action file.

- Control OpenFlow Schedule Execution

This service takes as input the id of the running Schedule and a specific type of command which can be resume, pause, stop or cancel the execution. Pause and resume options are mutually dependent as they require the schedule to be running or stopped. Cancel option stops the active Tasks and actions. These options are exposed in the OpenFlow User Interface through buttons and are presented in Figure 9 and Figure 10.

2.2.10. Request Replanning

OpenFlow has the ability to communicate with the AI Task Planning module to request the replanning of a Schedule that is paused either by the user or triggered by an event that OpenFlow had to stop the execution. Replanning can be invoked only on a stopped a Schedule.

OpenFlow invokes the Task Planning ROS Action Server of AI Task Planning module which uses the remaining unfinished Tasks and available Resources to create a new Schedule as described in D1.4. Once OpenFlow retrieves the new Schedule, it stores it in Knowledge Repository as an available Schedule for execution. Afterwards Schedule is loaded for execution and displayed in OpenFlow UI as a new Schedule. User can take any actions offered by OpenFlow on this Schedule as on any other regular Schedule. Figure 3 presents the data exchange between OpenFlow and AI Task Planning module.

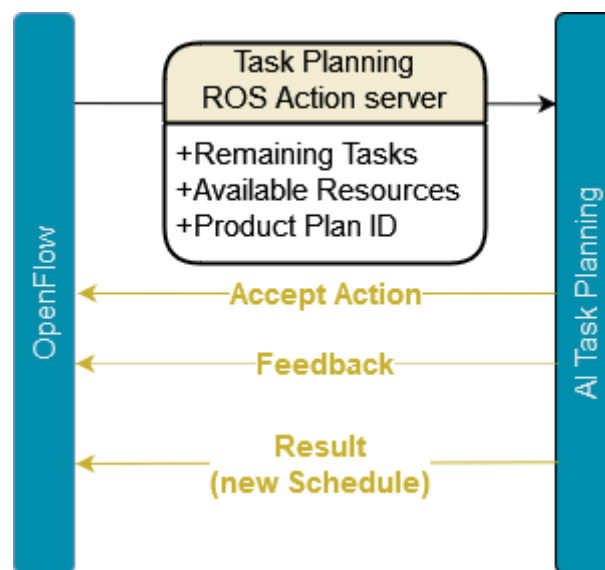


Figure 3: OpenFlow: Request Replanning

The development of the AI Task Planning advanced functionality is currently work in progress. The first set of designed interfaces have been developed and are included in the OpenFlow initial prototype.

2.2.11. Validate Open Schedules

OpenFlow can validate Schedules in Knowledge Repository before their execution to ensure their integrity and cohesion. As described in 2.2.7, OpenFlow uses the Actor model to manage

the actuation of Schedule's tasks and actions. In order for the Actor responsible to execute the Schedule to be created, this validation check is required. The validation can inform about syntactic errors and prevent the execution of erroneous schedules. The validation functionality not only informs that there are errors but also provides some information that help identify the source of the error.

A Schedule consists of many interconnected metadata residing inside Knowledge Repository such as its Tasks, Actions and Resources, of which are often referenced during execution through their unique identifier ID. Some of the validations and verification checks are the following:

- Verify that all collections of Actions, Resources and Tasks of a Schedule are populated and not empty.
- Verify that the Actions identifiers IDs for each set of next Actions to execute, can be identified and sourced to actual Actions that exist in KR.
- Verify that the Actions identifiers IDs which are part of a Task, can be identified and sourced to actual Actions that exists in KR.

2.2.12. OpenFlow Knowledge Repository

OpenFlow Knowledge Repository (KR) module is a submodule of OpenFlow and is responsible for modelling and maintaining the required information for OpenFlow core functionalities. This information ranges from Users and Product Plans to Schedules, Resources and network interface definitions. KR is designed following the Domain Driven Design (DDD) whereas for each OpenFlow specified domain context it offers factories, repositories and services for efficient Create, Read, Update, Delete (CRUD) operations of domains [11]. KR utilizes MongoDB to implement the aforementioned functionalities.

Figure 4 shows part of the data Model implemented in KR for Users, Product Plans and Schedules for execution.

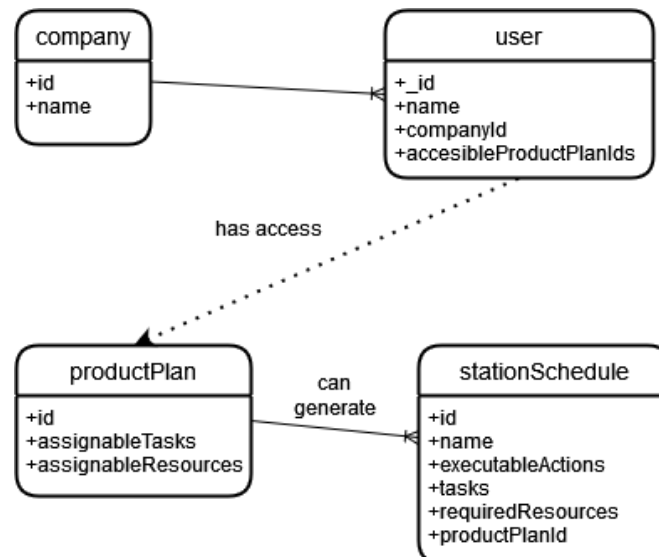


Figure 4: OpenFlow KR: User, Product Plan, Schedule Data Model

Additionally, Knowledge Repository maintains and imports the datasets required for all ODIN Pilot Cases. Such datasets include among others the required tasks, resources and suitabilities

following the architecture in D1.4. In the scope and implementation of current deliverable the preliminary White Goods and Automotive M18 pilot cases are persisted in KR.

For instance, the set of Tasks persisted in Product Plan of the preliminary White Goods pilot case, following the requirements specified in D1.1 and adjustment iterations following the agile approach, are presented in Table 6 below.

Table 6: White Goods preliminary demo tasks

White Goods Preliminary demo Tasks
Get Parallel Gripper
Pick Knob from kitting table
Place Knob to assembly table
Leave Parallel Gripper
Get Magnetic Gripper
Pick Transformer from kitting table
Place Transformer to assembly table
Pick Medium Cooktop from kitting table
Place Transformer in the oven
Place Medium Cooktop assembly table
Pick Big Cooktop from kitting table
Place Big Cooktop to assembly table
Pick Small Cooktop from kitting table
Place Small Cooktop to assembly table

These Tasks, alongside with their Resources and Actions are used to generate an execution Schedule which is stored in Knowledge Repository too [8]. Furthermore, Knowledge Repository stores the Action models described in 2.2.7.

2.2.13. Information Exchange with ERP systems

For the seamless integration of ODIN OpenFlow in a manufacturing environment, the need to communicate and receive product and resources information from external software systems has been described in D1.5. Such systems include ERP, PLM, MES and SCADA.

OpenFlow has the ability to connect with an external ERP system based on SAP and receive required information for production orders. This feature is currently under development and during WP4 through M18, a connection has been established with the ERP system of AEROTECNIC utilizing Java interfaces of SAP Java Connector module.

Information through the established connection included quantity, production number and due time of a production order and the data OpenFlow will be able to share with ERP consist of order status updates and an estimation of the expected successful completion of the order. Additionally, data retrieved would include the locations on the shopfloor a Fan Cowl has to be transported by mobile robots during its order execution. Figure 5 below presents the aforementioned information exchange with the ERP system.

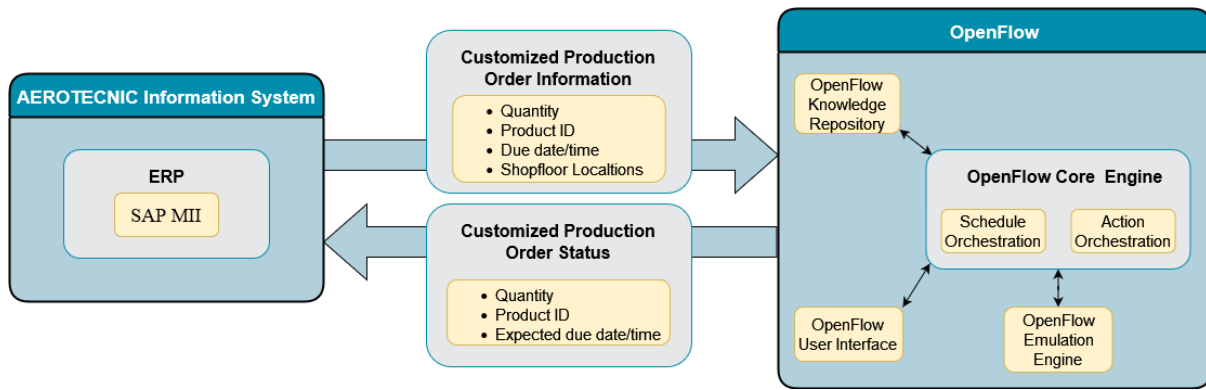


Figure 5: Information Exchange with AEROTECNIC - SAP

2.2.14. User Interface

OpenFlow User Interface (UI) is the central gateway between the OpenFlow and the user. It offers a user-friendly environment to control, monitor and view information such as the available Product Plans, Execution Schedules and Resources.

The OpenFlow UI consists of five main Tab pages that provide functionality related to the concepts described above. Table 7 below presents the navigation tabs in the OpenFlow UI main page.

Table 7: OpenFlow UI: Navigation Tabs

Tab Name	Description
Execution Status	Monitoring and controlling execution of Schedules
Schedules	Schedule selection & information
Product Plans	Create Schedules from available Product Plans
Resources	Available Resources and Network Resources

2.2.14.1. Login Page

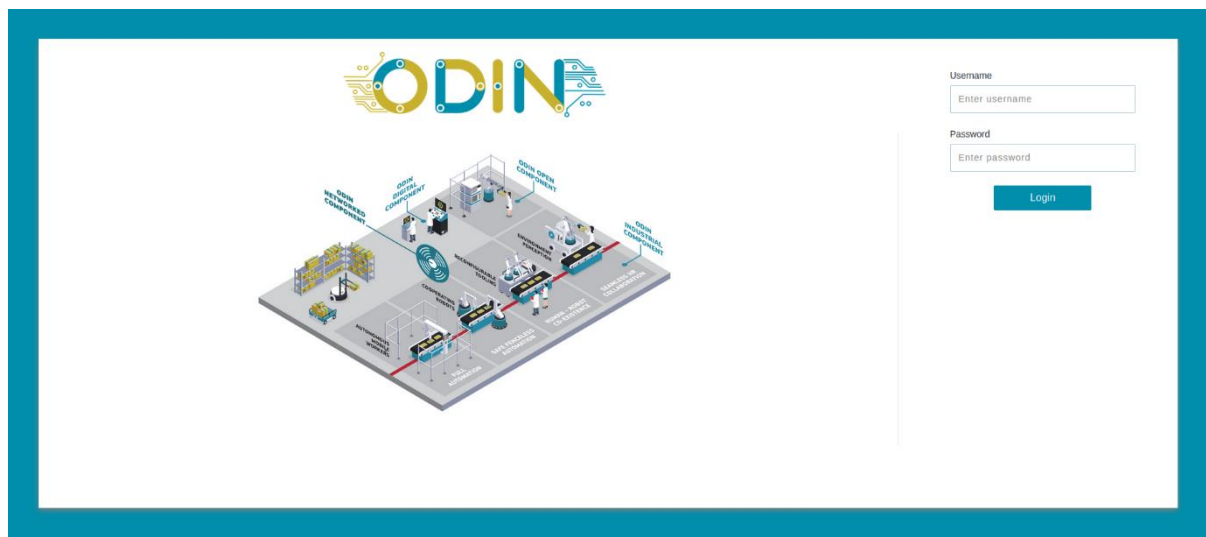


Figure 6: OpenFlow UI: Login Page

Login Page offers a user-friendly, simple login functionality to OpenFlow UI. Currently users belong to one company and can only view the information of the company they belong to. One company per pilot case has been created. Upon logging in, the OpenFlow redirects the user to

the Schedules Tab. The user can create new Schedules by navigating to the Product Plans tab and select to plan a new Schedule as described in section .

2.2.14.2. Schedules Tab

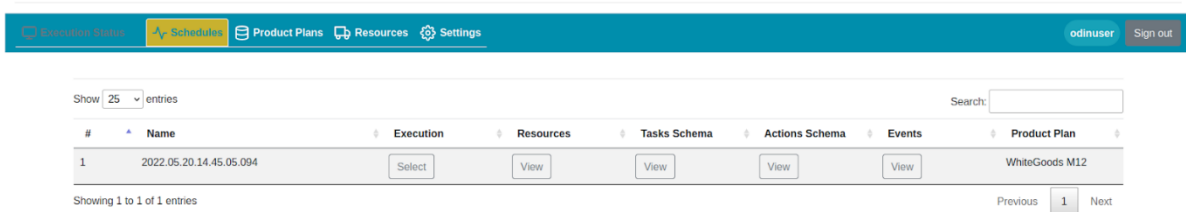


Figure 7: OpenFlow UI: Schedules Tab

The Schedules Tab displays all the available schedules that are visible to the user. A Schedule can be generated from the AI Task Planner for a specific Product Plan. The Schedules Tab offers access to visual representations of the required Resources of each Product Plan and information about these Resources, Schedule’s Tasks and Actions and Events. Finally, this tab offers the functionality to select a Schedule for execution by clicking the related “Select” button in the Execution column.

2.2.14.3. Execution Status Tab

The OpenFlow UI Execution Status Tab provides information about a running schedule and offers control options to the user, effectively allowing to control the production execution. The Execution Status Tab is only enabled if a schedule is selected for execution. For instance, the Execution Status Tab is not enabled in Figure 7. In order to see the status tab, the user needs to select a Schedule as described in section 2.2.14.2. Selecting a Schedule will enable and redirect to the new Execution Tab shown in Figure 8, that shows an instance of the enabled Schedule Status Tab.

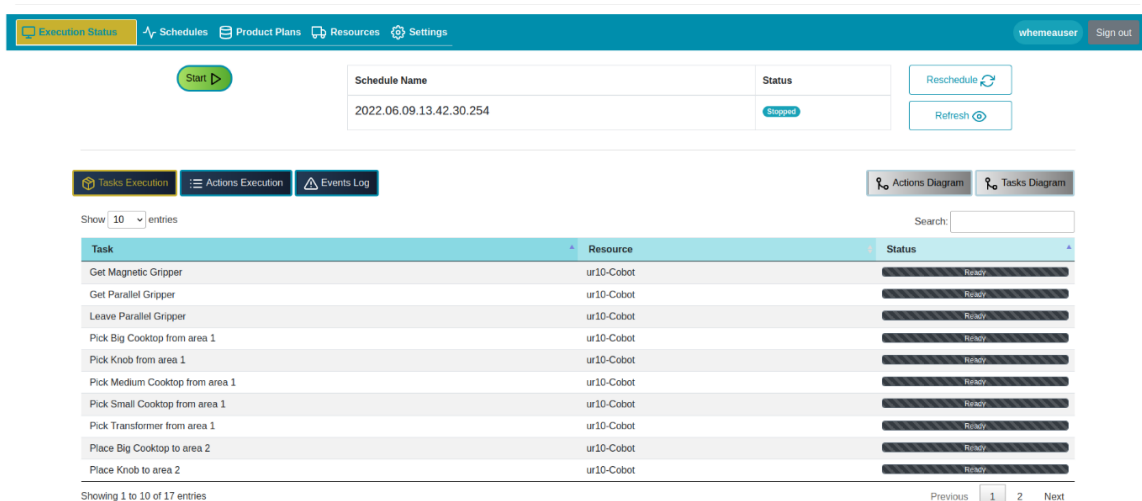


Figure 8: Open Flow UI: Execution Status Tab

The OpenFlow UI Execution Status Tab enables the control of a Schedule and offers visualization of the execution in Task level or in the Action level which is more detailed.

- **Controlling the execution of Schedule.**

Start button starts the execution of the selected Schedule and enables the Pause-Resume & Cancel buttons.

While the Schedule is running, the User has the option to pause the current execution and resume it later on demand.



Figure 9: Options while Schedule is running

The Pause button is replaced by the Resume option if the Schedule is already paused as shown in Figure 9. The Cancel Button cancels the Tasks & Actions that are active. The UI Controls are then updated to only allow the user to Start the Schedule from the first Task again.



Figure 10: Options while Schedule is paused

Additionally, when the Schedule is paused or stopped, the User can reschedule the remaining Tasks as a new Schedule ready for execution through the Reschedule button in Figure 8.

- **Schedule Status Visualization functionality**

The Execution Status Tabs offers real time monitoring on the status of Tasks & Actions as well as visualizing the Schedule in a Graph.

- o **Task Execution Status**

The screenshot shows a web interface with three tabs: 'Tasks Execution' (selected), 'Actions Execution', and 'Events Log'. On the right, there are two icons for 'Actions Diagram' and 'Tasks Diagram'. Below the tabs, there is a search bar and a 'Show 10 entries' dropdown. The main content is a table with three columns: 'Task', 'Resource', and 'Status'. The first task, 'Place Medium Cooktop to area 2', is currently 'Active' (indicated by a blue progress bar). Other tasks are in 'Ready' (grey bars) or 'Fished' (green bars) states. At the bottom, it says 'Showing 1 to 10 of 17 entries' and has navigation buttons for 'Previous', '1', '2', and 'Next'.

Task	Resource	Status
Place Medium Cooktop to area 2	ur10-Cobot	Active
Pick Big Cooktop from area 1	ur10-Cobot	Ready
Pick Small Cooktop from area 1	ur10-Cobot	Ready
Place Big Cooktop to area 2	ur10-Cobot	Ready
Place Small Cooktop to area 2	ur10-Cobot	Ready
Program a new robot task	Operator	Ready
Stop	Controller	Ready
Get Magnetic Gripper	ur10-Cobot	Fished
Get Parallel Gripper	ur10-Cobot	Fished
Leave Parallel Gripper	ur10-Cobot	Fished

Figure 11: Tasks Execution Status

While the Schedule is running, the status of the Tasks is constantly updated to show the actual execution status. The Tasks consist of many actions and offer a higher level of abstraction and observation. The screenshot in Figure 11 shows tasks in different status at the same screen.

The screenshot shows a web interface with three tabs: 'Tasks Execution', 'Actions Execution', and 'Events Log'. The 'Actions Execution' tab is active. At the top right, there are buttons for 'Actions Diagram' and 'Tasks Diagram'. Below the tabs, there is a search bar and a 'Show 10 entries' dropdown. The main content is a table with the following data:

Task	Action	Resource	Status
Place Small Cooktop to area 2	ImmutableMoveJointsAction-dcadf6b8-cb68-49f2-9f3d-eba041b261f9	ur10-Cobot	Ready
Program a new robot task	Program a new robot task	Operator	Ready
Program a new robot task	Sync All	Operator	Ready
Stop	Stop Schedule	Controller	Ready
Stop	Sync All	Controller	Ready
Get Magnetic Gripper	ImmutableMoveJointsAction-061c034e-c431-4c62-864f-5c47057a0f20	ur10-Cobot	Finished
Get Magnetic Gripper	ImmutableMoveJointsAction-2c005e35-5e3f-47c5-86a6-3726237f1154	ur10-Cobot	Finished
Get Magnetic Gripper	ImmutableMoveJointsAction-86344ae5-dc21-4f12-94bb-e83cb327f6c8	ur10-Cobot	Finished
Get Magnetic Gripper	ImmutableMoveJointsAction-81a8fbb-2e68-4f2f-a944-c14c4e4092ad	ur10-Cobot	Finished
Get Magnetic Gripper	ImmutableReferenceExecutionAction-c3e1cef7-f31b-4ea6-89a0-de2fc8bf7e6e	ur10-Cobot	Finished

At the bottom of the table, it says 'Showing 1 to 10 of 108 entries'. To the right, there is a pagination control with 'Previous', '1', '2', '3', '4', '5', '...', '11', and 'Next'.

Figure 12: Actions Execution Status

o **Action Execution Status**

Similar to the Task Execution Status, the Actions Execution tab panel displays actions names, their resources and Task and their status as shown in Figure 12. The Actions are atomic execution steps that are part of a higher-level Task abstraction.

o **Actions & Tasks diagrams**

The OpenFlow UI can graphically depict the Execution Schedule in either Action or Task level granularity. Actions and Tasks diagram option in Execution Status Tab offers visualization of the sequence of required actions and tasks respectively in order to execute a complete Schedule. A new diagram is generated for each case and a new page will load displaying the respective graph. Figure 13 shows the task diagram of the preliminary White Goods pilot case. Due to the size of the diagram a zoomed in part has been added.

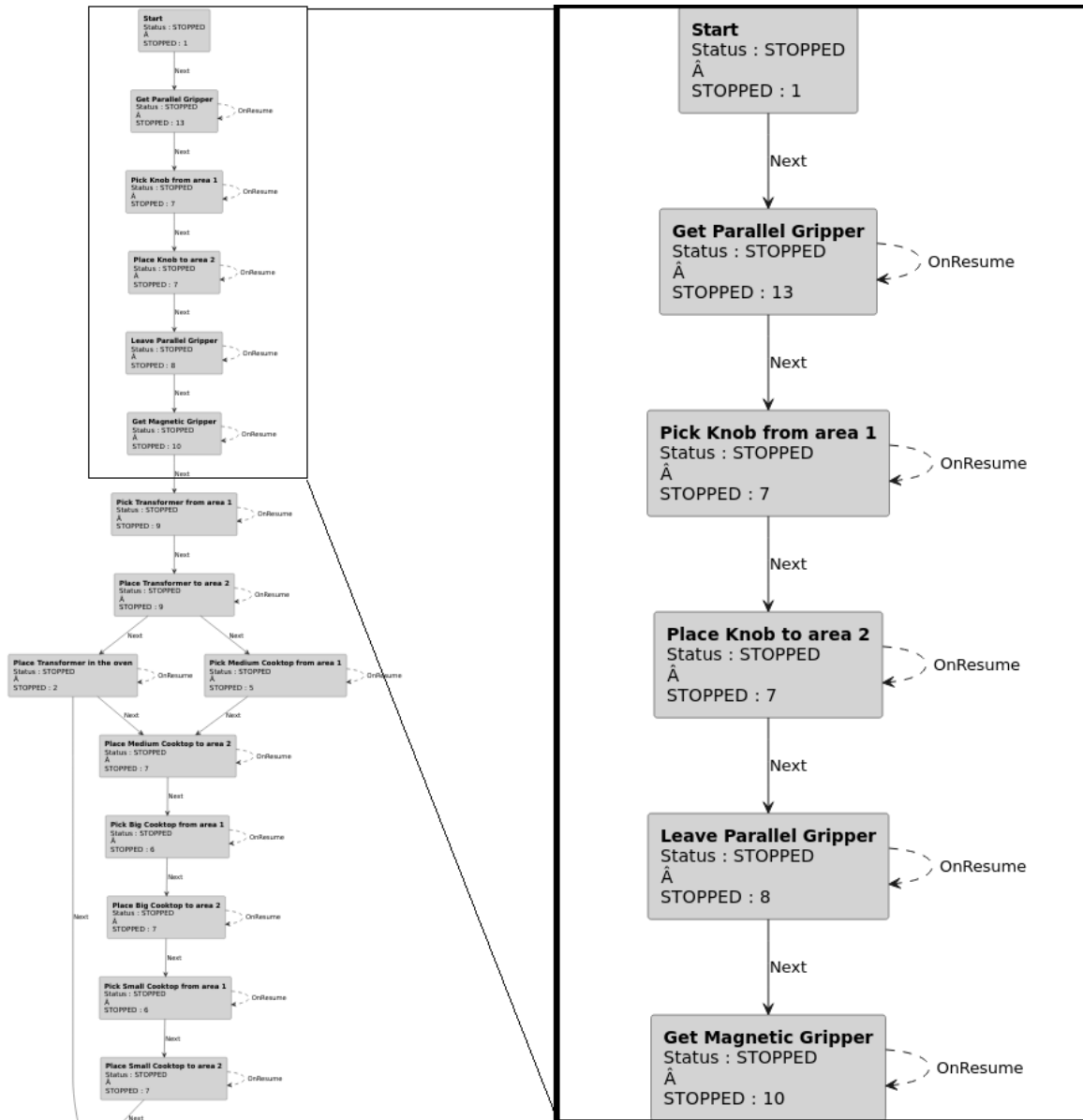
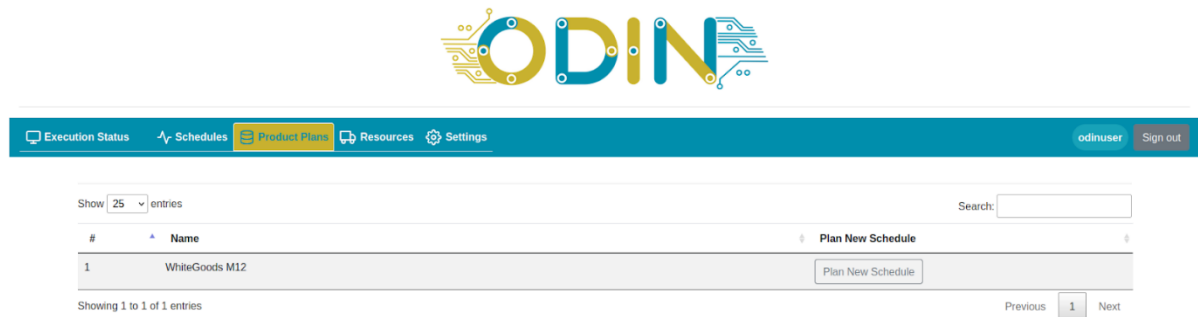


Figure 13: White Goods preliminary pilot case demo - Tasks diagram

2.2.14.4. Product Plans Tab

Product Plans tab displays the available Product Plans to a logged in User, which as described in 2.2.12. maintain the necessary data that can be used to generate a Schedule. Figure 9 shows the available product plans for White Goods pilot case. Each Product Plan can be used to generate a Schedule through the Plan New Schedule option.



The screenshot shows the ODIN Open Flow UI interface. At the top, there is a navigation bar with tabs for Execution Status, Schedules, Product Plans (selected), Resources, and Settings. The user is logged in as 'odinuser' and can click 'Sign out'. Below the navigation bar, there is a search bar and a 'Show 25 entries' dropdown. The main content area displays a table with the following data:

#	Name	Plan New Schedule
1	WhiteGoods M12	Plan New Schedule

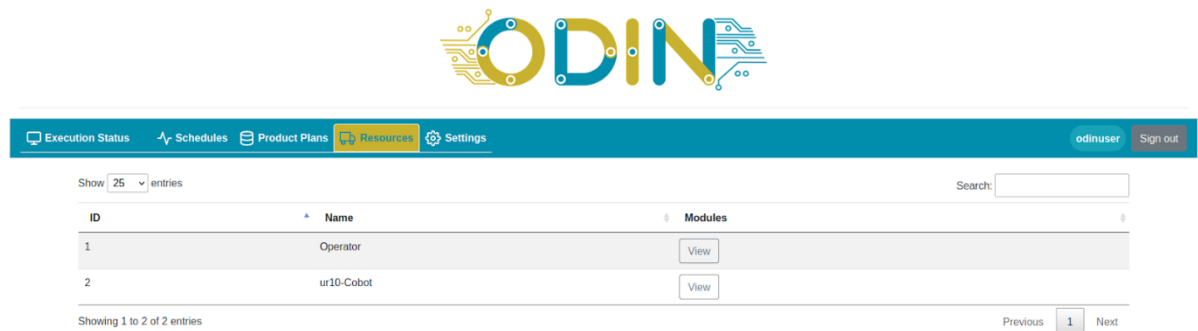
At the bottom of the table, it says 'Showing 1 to 1 of 1 entries' and 'Previous 1 Next'.

Figure 14: Open Flow UI: Product Plans Tab

If the User logs into the OpenFlow UI for the first time, he has to create a Schedule through Product Plans tab.

2.2.14.5. Resources Tab

Resources Tab displays all available resources to the company the User belongs to, which can be assigned as resources in Tasks and Actions of Schedules. Figure 15 shows the Resources of White Goods pilot case as described in 2.2.12.



The screenshot shows the ODIN Open Flow UI interface with the Resources tab selected. The navigation bar is the same as in Figure 14. The main content area displays a table with the following data:

ID	Name	Modules
1	Operator	View
2	ur10-Cobot	View

At the bottom of the table, it says 'Showing 1 to 2 of 2 entries' and 'Previous 1 Next'.

Figure 15: Open Flow UI: Resources Tab

Additionally, *Modules* column on each resource displays the available Network Resources that can be utilized from this resource in a Schedule. Figure 16 shows the available ActionLib Servers of ur10-Cobot for White Goods use case.

Sub-modules of this resource ×

Show entries Search:

#	Name
1	gripperControlGripperActionLibServerId
2	toolChangerControlToolChangerActionLibServerId
3	ur10MoveArmJointActionLibServerId
4	ur10ConfigureTcpActionLibServerId
5	ur10MoveArmCartesianActionLibServerId
6	controlScheduleExecutionActionLibServerId
7	NoOperation
8	ur10ConfigurePayloadActionLibServerId

Showing 1 to 8 of 8 entries Previous Next

Figure 16: Open Flow UI: Resource's modules

2.3. Design

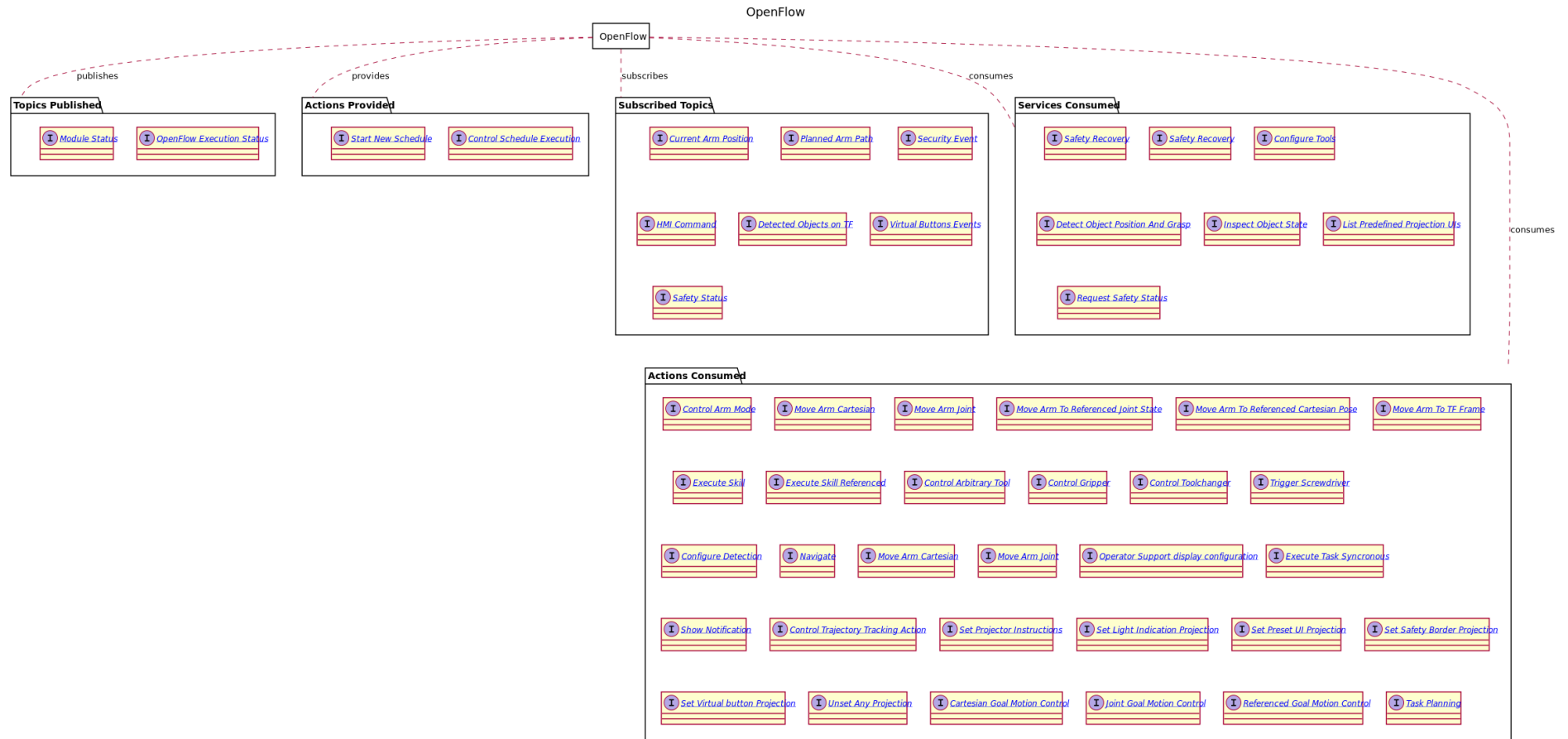


Figure 17 : OpenFlow Initial Prototype Interfaces Design

Figure 17 presents the implemented Actions Interfaces that OpenFlow orchestrates to offer the features described in section 2.2 adhering to the architecture defined in D1.4. OpenFlow structure facilitates adding interface connections. This modular structure allows to future-proof the scalability and extensibility of OpenFlow to adjust on any new requirements that are going to emerge through the development phase of WP4.

The interfaces of Figure 17 are available to use into the production Schedules of the described Pilot Cases for testing and further development. For instance, for the preliminary White Goods Pilot case, OpenFlow had to orchestrate the actuation of Actions of a Cobot robotic system. Figure 18 below presents the design of the implemented system for the preliminary Pilot Case which besides the Cobot had to manage interfaces for Operator’s Actions too.

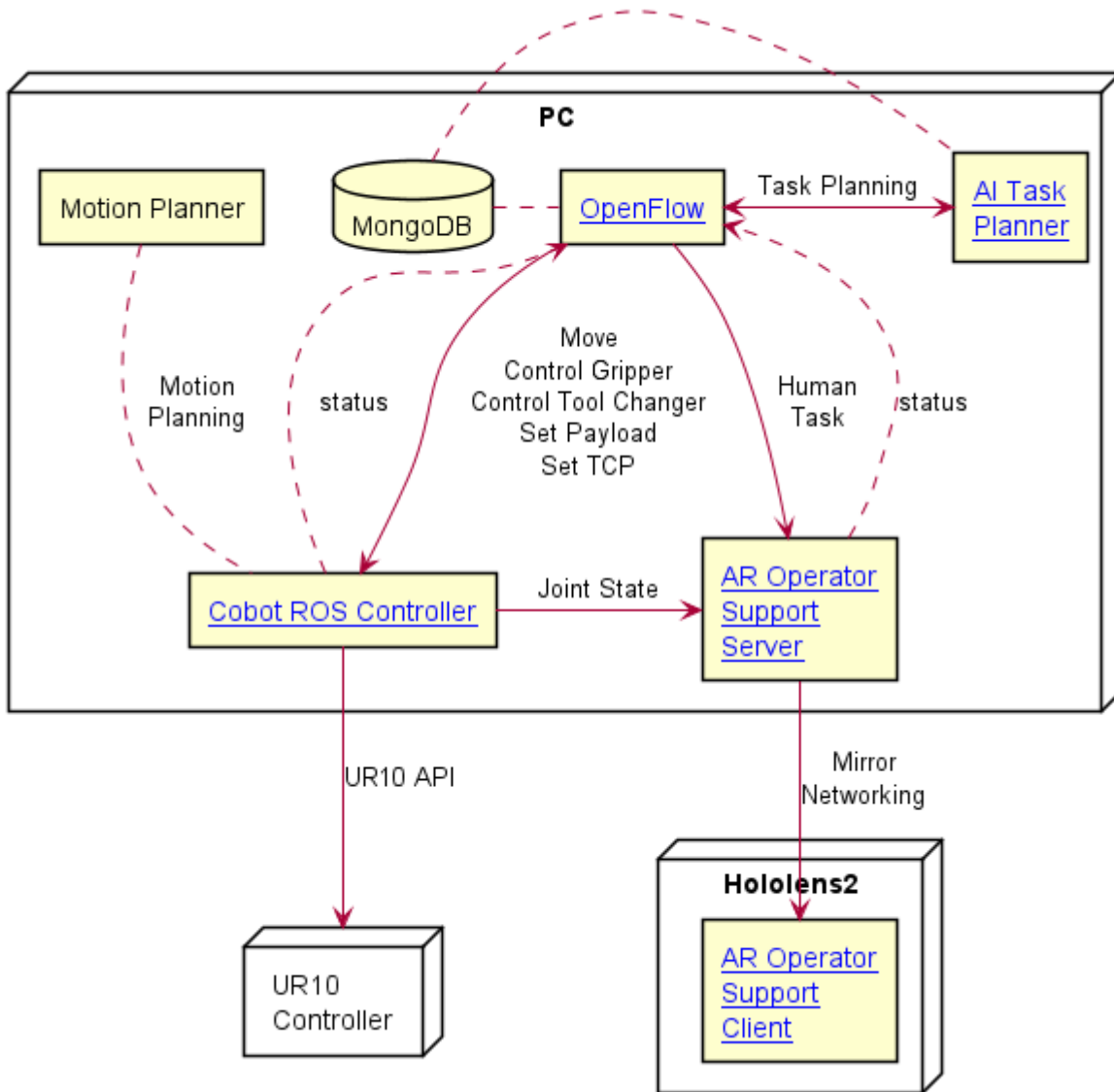


Figure 18: ODIN Preliminary White Goods Pilot Case deployment diagram

2.4. Initial Prototype

OpenFlow module is developed in Java and for storage purposes and data persistence MongoDB is used. Data maintenance follows the repository pattern to accomplish the functionalities of OpenFlow Knowledge Repository described in 2.2.12.

ROS interfaces for OpenFlow Core described in 2.2.7, 2.2.10 are developed using the ROS Java library. To manage the interfaces and interact with the “actuator” components for the interfaced Actions, OpenFlow uses the AKKA framework to implement the Actor model described in 2.2.7.

The OpenFlow is designed to be distributable through docker [9] container images. The OpenFlow initial prototype has been released as a docker image that allows all the partners in the consortium to use and evaluate the latest development state.

Prior to the M18 release that coincides with the D4.1 submission date a preliminary release was also provided to enable an early, preliminary White Goods integrated scenario that took place in LMS.

The OpenFlow docker images have been uploaded in the project’s private docker image repository that is hosted by LMS. ODIN project partners can request credentials and gain access to OpenFlow docker images.

3. CYBERSECURITY

This section describes the ODIN Cybersecurity module, in terms of its main features of threat modelling, detection and response.

3.1. Introduction

When speaking of protecting an environment, we need to divide the process in different steps:

- Threat modelling: understand how the attack can be performed.
- Detection: identify how to identify if an attack is ongoing.
- Response: prepare the defensive actions that will allow the minimization of the damage

In complex interconnected systems with relative visibility, this can be a difficult process to organize. A robotic environment in a factory is providing an increasing interconnected functionality, and thus, a growing attack surface.

To organize the protection of these environments it is needed to adapt the steps to the needs.

The main focus of this task is to study reference frameworks on how a cyberattack is performed, in order to adapt them to an environment as the one described in the ODIN project and to model possible threats accordingly. In particular, the modelling and protection will be focused on the scope of the ODIN Networked component.

Once the kind of threats and attacks to expect will be known, a specific way of searching for the referred attacks will be proposed by exploring traces in the systems that indicate some kind of offensive action.

The last step is to provide a way of responding to the detected threats, by signaling it, launching defensive actions or starting a customized treatment request in the operation center.

This way, the impact of threats that may occur in the system can be minimized.

3.2. Features

The main ODIN proposal for the Cybersecurity solution is based on three main features that provide an approach for the management of the environment:

- Threat modelling and attack surface definition,
- Detection,
- Response.

3.2.1. Threat modelling and attack surface definition

According to [10], attack surface Analysis is about mapping out which parts of a system need to be reviewed and tested for security vulnerabilities. Attack Surface Analysis tries to understand the risk areas in an application, to identify which parts of the application are open to attack, to find ways of minimizing this, and to notice when and how the Attack Surface changes but also what this means from a risk perspective.

According to [12], threat modelling is a structured approach of identifying and prioritizing potential threats to a system and determining the value that potential mitigations would have in reducing or neutralizing those threats.

Once the attack surface is identified, an adapted threat modelling can be performed.

For this aim, MaGMa [16] and MITRE ATT&CK [13] methodologies have been analyzed. They are actually closely related one each other.

MaGMA is a Use Case Framework, created collaboratively by several Dutch financial institutions. The main element of the security management is the use case, which MaGMA defines as “a security monitoring scenario that is aimed at the detection of manifestations of a cyber threat”. The use cases are subdivided in three levels, from the top Business layer, describing how it is connected to the organizational needs, Threat layer, describing how the use case can be menaced and the low-level Implementation layer where the technical and operational aspects of the architecture are described. The threats are also divided in three levels of detail, from higher L1, L2 (both being part of the Threat/tactical layer) and the actual monitoring rules covered in the L3 level, based on the MITRE ATT&CK Matrix for Enterprise. This structure is the way of linking a top-level business view to a low-level technical asset or operation.

MITRE ATT&CK is a framework that aims to document the common tactics and techniques used against IT and OT environments. This framework divides an attack in different phases, called tactics, that are performed in sequence, although not all may be necessarily used, in order to complete a successful cyberattack. Each one of the steps can be performed with a catalogue of adversarial techniques, offensively oriented actions against the platform.

These approaches will be followed for the attack modelling methodology definition. The proposed attack model will include a set of techniques that can be used for a hypothetical attack to the ODIN platform.

In section 3.3.1 the attack modelling methodology will be described, and in section 3.4.1, a concrete attack surface and threat model for ODIN will be developed.

3.2.2. Detection

In cybersecurity, detection is the ability to search for traces and identify possible attacks. The implementation of detection will be based on Security Information and Event Management (SIEM) tools, with the following capabilities:

- Event ingestion:
 - o Collection of raw data from the network and systems.
- Event generation:
 - o Normalization,
 - o Aggregation,
 - o Correlation.

In section 3.3.3 the Cybersecurity solution architecture, that includes the SIEM component, is presented, while section 3.4.3 provides further information of the SIEM implementation and deployment.

3.2.3. Response

In cybersecurity, response is the ability to orchestrate the defensive actions when a possible attack is identified. The proposed implementation of response actions will be based on Security Orchestration, Automation and Response (SOAR) tools, with the following capabilities:

- Workflow,
- Automation,
- Incident response,
- Ticketing and communication.

In section 3.3.3 the Cybersecurity solution architecture, that includes the SOAR component, is presented. Further information of the SOAR implementation and deployment are described in section 3.4.3.

3.3. Design

This section describes the cybersecurity system for ODIN, according to the features described in section 3.2, and how it is integrated in the ODIN architecture. For that, first the attack modelling methodology used for the project is identified and described. Then the ODIN use cases architecture is presented, and the scope and concrete tools are defined which will be deployed for cybersecurity. Finally, the validation process is described.

3.3.1. Attack modelling methodology

When defining and testing cybersecurity in an environment, the main actors are used:

- Red Team. Plays the role of the attacker. The goal would be to identify a sequence of actions to gain access to the main goal. This sequence is called a Kill Chain [15] and it's composed of different techniques in all or some of the tactics described in the MITRE ATT&CK model.
- Blue Team. Plays the role of the defender. Should be aware of the attack surface provided by the system in order to protect it. Its goal is to identify all possible techniques in each tactic to which it is exposed and protect them, so no Kill Chain can be found by an attacker.

These roles represent the two positions in a cyberattack, attacker and defender, and reflect the main difference in what success means for each one. While an attacker just needs one successful path, namely Kill Chain, to achieve the goal, the defender needs to assure and protect a wide range of possible steps that may be used to gain access to the target.

Therefore, the methodology approach for a defender is to be aware of where the attacker may advance in the chain of actions that will lead to an intrusion.

- Identify all techniques that apply to the architecture.
- Describe how to exploit the adversarial technique.
- Identify how to detect an attack with that technique (Detection Technique).
- Implement detection techniques.
- Describe how to react on an attack (Protection Technique).
- Implement protection techniques.

3.3.2. ODIN network architecture

This section presents the network architecture for ODIN. The ISA 99 reference framework [14] has been analyzed, that aims at including security in the design of industrial networks and adapting it to the ODIN network architecture. Over this scenario, the scope of the cybersecurity detection and response, and where the different elements are placed on this architecture will be defined in section 3.3.3.

The following figure presents the existing state of the ODIN Network Architecture.

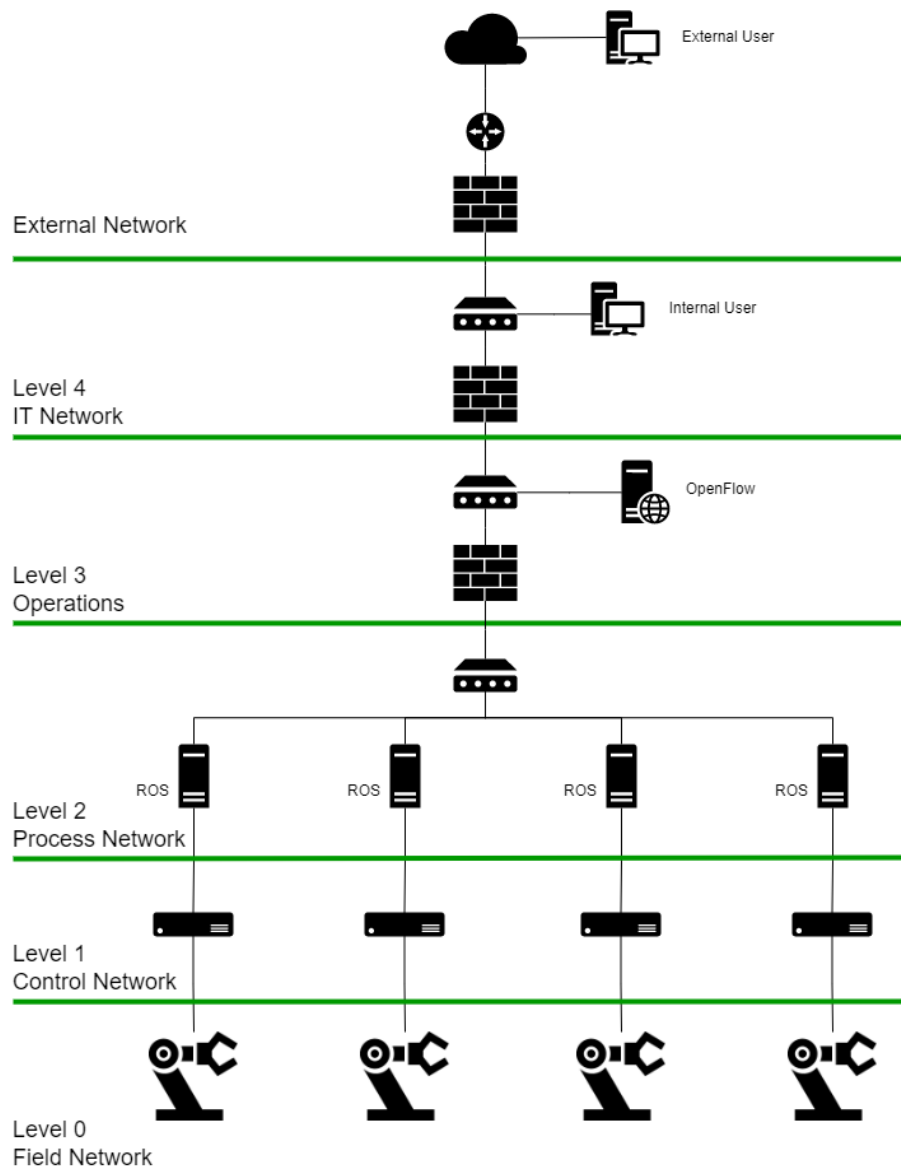


Figure 19: ODIN Network Architecture based on IEC 62443 / ISA99 model

This proposed scenario presents a network architecture segmented in 5 levels with segregation according to IEC 62443 [17] and ISA 99 to isolate different IT and OT networks.

- Level 0 – Field Network: The different Robots are located at this level.
- Level 1 – Control Network: The Robot Controllers are located here.
- Level 2 – Process Network: The ROS Controllers are placed at this level.
- Level 3 – Operations: OpenFlow is located here at the existing state.
- Level 4 – IT Network: IT services are located at this network.

3.3.3. Cybersecurity Module Architecture

Over the ODIN network architecture defined in section 3.3.2, this section describes where to locate the different components of the ODIN Cybersecurity module, and which is the scope of the cybersecurity protection.

On the one hand, the main elements of the ODIN Cybersecurity system are:

- SIEM for collection of raw data from the network and systems and event generation.
- SOAR for security response and orchestration.

On the other hand, according to the scope of task T4.2, the cybersecurity monitoring and protection is focused on the scope of the ODIN Networked component targeting on the modelling, detection and response features of the ODIN Cybersecurity module integrated with the OpenFlow implementation.

The following figure shows the ODIN Network architecture with cybersecurity tools (SIEM and SOAR) integrated.

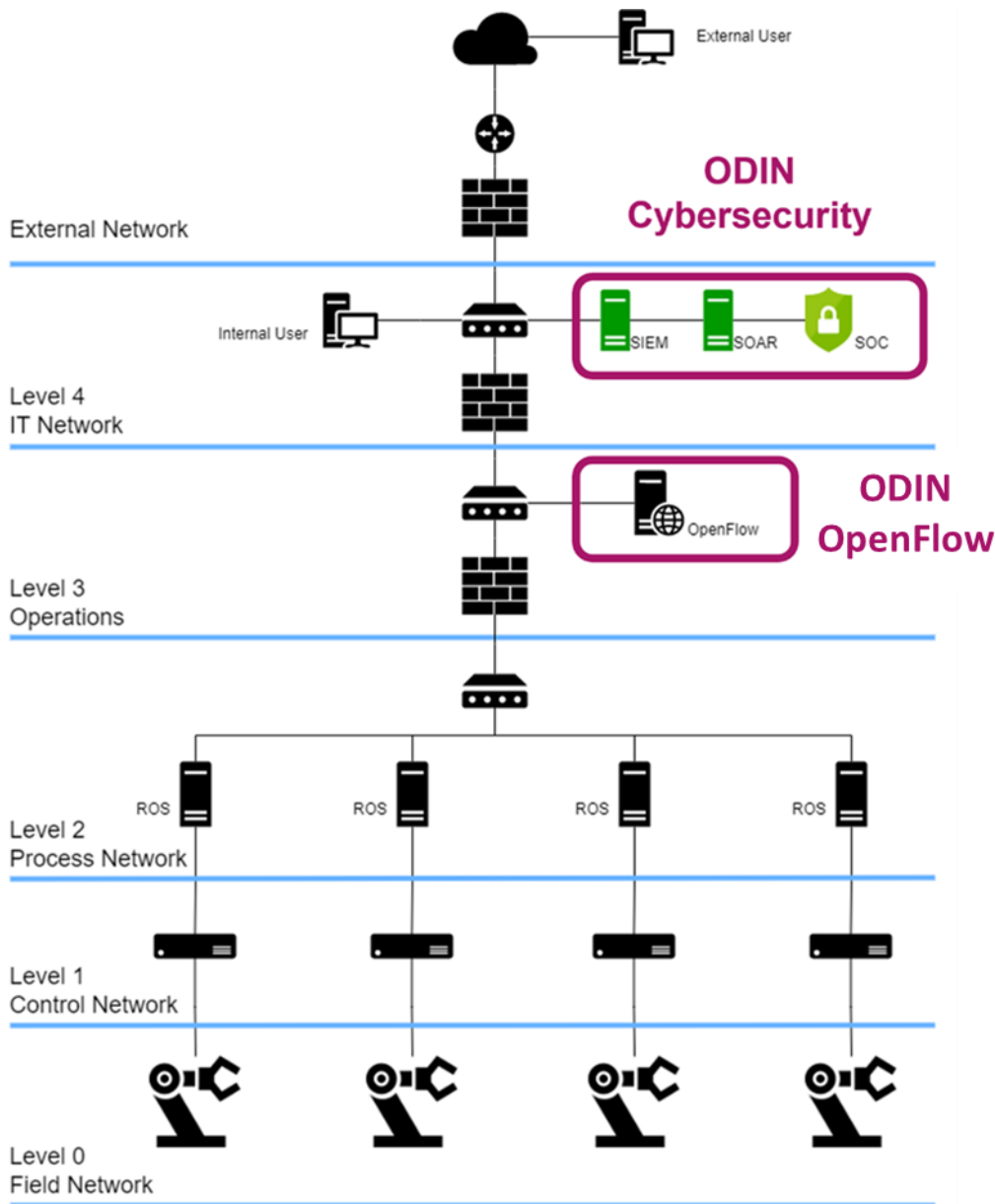


Figure 20: ODIN Network Architecture with Cybersecurity tools integrated

In the figure above, ODIN Cybersecurity System is located at Level 4, in the IT Level. It provides detection and protection services over the ODIN OpenFlow module, which was initially identified at Level 3, Operations Level.

3.3.4. Validation

The validation of the Cybersecurity system will be done through the following steps:

1. Test individual attacks that use attack techniques.
2. Identify complete Kill chains that leads to a use case compromise by linking different adversarial techniques. Test sequence.
3. Detect individual attacks with detection techniques.
4. React using protection techniques.

3.4. Initial Prototype

This section presents the initial prototype for the ODIN Cybersecurity system. For this aim a concrete attack scenario and threat modelling for the use case is defined. Then, the prototype environment is described, and finally the Detection and response solutions are presented.

It is important to define here the assumptions of the prototype, in terms of defining what is out of the scope of the project in terms of attack surface definition and threat modelling.

3.4.1. Prototype environment

This section describes the working environment that is being used for ODIN Cybersecurity System prototype design and implementation. As mentioned before, the target of the security services is the ODIN Networked component. Therefore, it is necessary to deploy this component in the working environment to integrate it with Cybersecurity solution.

For the prototype environment, the emulation mode of OpenFlow will be used, that will provide the digital twin of the networked component. Over this emulated scenario, the cybersecurity system will be deployed and several cybersecurity exploits and tests, without affecting the real deployment will be deployed.

Based on Figure 18 which describes the OpenFlow module deployment in a real scenario, the emulated environment is described in Figure 21.

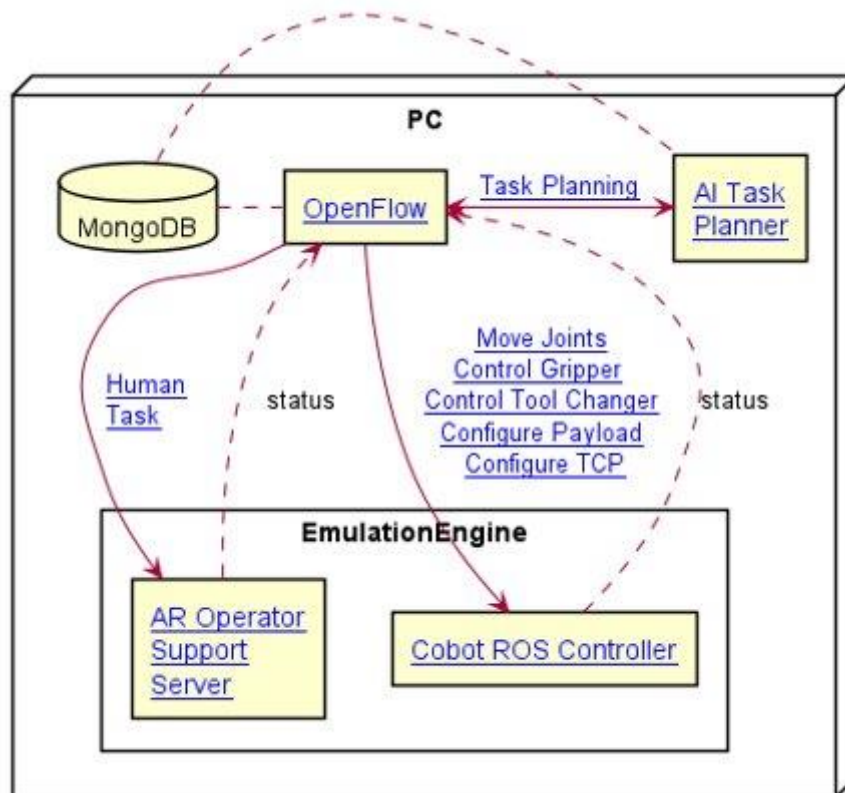


Figure 21: ODIN Networked Component Emulation

The emulation of the ODIN Networked component provides the following elements:

- OpenFlow instance: provided as a docker image.
- ROS Core: a toolkit that interacts with OpenFlow.
- MongoDB: OpenFlow database.

This emulation is provided by the project through a containerized format and the details about the implementation are the following:

- Flavour: ROS1.
- ROS Distro: Noetic.
- ROS Version: 1.15.14.

3.4.2. Attack scenario

The attack scenario will be defined through:

- The attack flow / steps.
- The attack surface.
- The attack model, with all the techniques and tactics.

The following figure presents the interactions flow that may occur over OpenFlow.

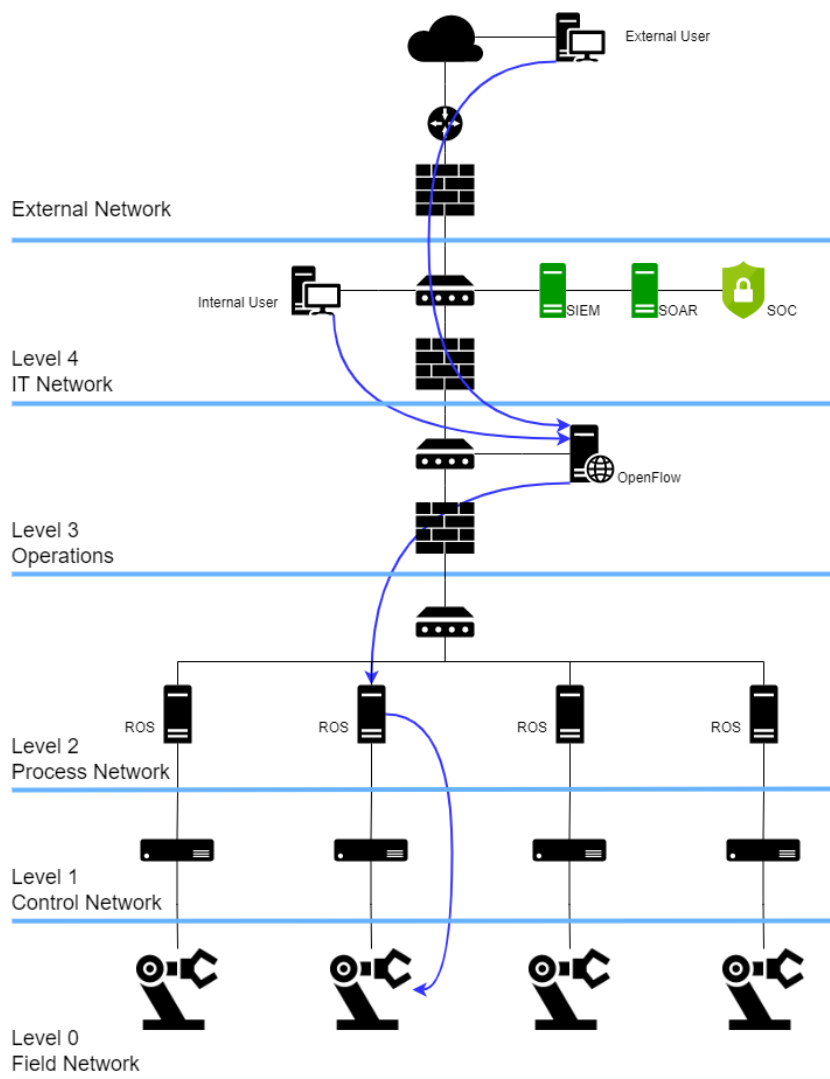


Figure 22: ODIN Architecture interactions flow

OpenFlow is managed through a Web UI, and according to the figure above, this interface could be accessed by internal (Level 4) and external users. Then, the planned operations will be forwarded from the OpenFlow (Level 3) to the ROS Controller (Level 2), so that they are forwarded again to the Robot Controller (Level 1) and finally to the Robot (Level 0).

Based on this interaction flow, the elements with which the OpenFlow interacts can be derived, and its interfaces and the attack surface can be identified.

- OpenFlow networked component.
- Host and OS environment.
- Docker environment.
- Web App environment.
- ROS communication (ROS1 and ROS2).

A list of available servers/publishers and clients/subscribers has been already provided in the GitHub repository and can be visualized in Figure 23.

Servers/Publishers

Search:

Name	Context	Graph Name	Clients/Subscribers	Server/Publisher	Technology	Current Target Version	Activity	Status %	Contact	Type	Use Case Scenarios
Configure Payload	execution	execution/cobot/integration/actions /configure_payload		cobot	ROS	M12 Demo	Design	50		Action	
Configure TCP	execution	execution/cobot/integration/actions /configure_tcp		cobot	ROS	M12 Demo	Design	50		Action	
Control Arm Mode	execution	execution/cobot/integration/actions /control_arm_mode	open_flow	cobot	ROS	1st Version	Design	50		Action	
Current Arm Position	execution	execution/cobot/integration/topics /current_arm_position	digital_twin open_flow operator_support projector_interface task_planning	cobot	ROS	1st Version	Design	50		Topic	
Move Arm Cartesian	execution	execution/cobot/integration/actions /move_arm_cartesian	open_flow	cobot	ROS	M12 Demo	Implementation	50		Action	
Move Arm Joint	execution	execution/cobot/integration/actions /move_arm_joint	open_flow	cobot	ROS	M12 Demo	Implementation	50		Action	
Move Arm To Referenced Cartesian Pose	execution	execution/cobot/integration/actions /move_arm_to_referenced_pose	open_flow	cobot	ROS	1st Version	Design	50		Action	
Move Arm To Referenced Joint State	execution	execution/cobot/integration/actions /move_arm_to_referenced_joint_state	open_flow	cobot	ROS	1st Version	Design	50		Action	
Move Arm To TF Frame	execution	execution/cobot/integration/actions /move_arm_to_tf	open_flow	cobot	ROS	1st Version	Design	50		Action	
Planned Arm Path	execution	execution/cobot/integration/topics /planned_arm_path	digital_twin open_flow operator_support projector_interface task_planning	cobot	ROS	1st Version	Design	50		Topic	

Figure 23: OpenFlow publishers and subscribers list

Once the attack surface for the security target is defined, the threat modelling will be defined, according to the methodology presented in section 3.3.1.

First, a Cyber Kill Chain for OpenFlow will be defined. A cyber kill chain for an automated robotics scenario like ODIN will take into account the IT network (external network) and OT network (internal network) where the phases of the kill chain will be the following:

- **Reconnaissance.** Research, identification, and selection of targets.
- **Weaponization.** Before attacking the target, the threat actor need to perform an effective way to perform the attack. Weaponization is the process where tools are built or used.

- **Cyber Intrusion.**
- **Privilege escalation.**

Knowledge of the Cyber Kill Chain allows operators and security officers to apply specific measures to this field aimed at protecting control systems at each stage of the chain, for instance, SIEM systems.

Secondly, MaGMA and MITRE ATT&CK frameworks will be adapted for ODIN by selecting the different applicable tactics. The MITRE Tactics are:

- Initial Access.
- Execution.
- Persistence.
- Evasion.
- Discovery.
- Lateral Movement.
- Collection.
- Command and Control.
- Inhibit Response Function.
- Impair Process Control.
- Impact.

In ODIN, from the Cybersecurity perspective, a threat scenario will be modelled for OpenFlow, which is composed by the followed components:

- Web application Front End:
 - Apache
 - Tomcat Embedded
- ICS environment
- Docker environment

Therefore, the MITRE Tactics can be filtered and adapted to these environments, as shown in the figure below:

Initial Access 12 techniques	Execution 9 techniques	Persistence 5 techniques	Privilege Escalation 2 techniques	Evasion 6 techniques	Discovery 5 techniques	Lateral Movement 6 techniques	Collection 10 techniques	Command and Control 3 techniques	Inhibit Response Function 13 techniques	Impair Process Control 5 techniques	Impact 12 techniques
Drive-by Compromise	Change Operating Mode	Modify Program	Exploitation for Privilege Escalation	Change Operating Mode	Network Connection Enumeration	Default Credentials	Automated Collection	Commonly Used Port	Activate Firmware Update Mode	Brute Force I/O	Damage to Property
Exploit Public-Facing Application	Command-Line Interface	Module Firmware	Hooking	Exploitation for Evasion	Network Sniffing	Exploitation of Remote Services	Data from Information Repositories	Connection Proxy	Alarm Suppression	Modify Parameter	Denial of Control
Exploitation of Remote Services	Execution through API	Project File Infection		Indicator Removal on Host	Remote System Discovery	Lateral Tool Transfer	Detect Operating Mode	Standard Application Layer Protocol	Block Command Message	Module Firmware	Denial of View
External Remote Services	Graphical User Interface	System Firmware		Masquerading	Remote System Information Discovery	Program Download	I/O Image		Block Reporting Message	Spoof Reporting Message	Loss of Availability
Internet Accessible Device	Hooking	Valid Accounts		Rootkit	Wireless Sniffing	Remote Services	Man in the Middle		Block Reporting Message	Unauthorized Command Message	Loss of Control
Remote Services	Modify Controller Tasking			Spoof Reporting Message		Valid Accounts	Monitor Process State		Block Serial COM		Loss of Productivity and Revenue
Replication Through Removable Media	Native API						Point & Tag Identification		Data Destruction		Loss of Protection
Rogue Master	Scripting						Program Upload		Denial of Service		Loss of Safety
Spearphishing Attachment	User Execution						Screen Capture		Device Restart/Shutdown		Loss of View
Supply Chain Compromise							Wireless Sniffing		Manipulate I/O Image		Manipulation of Control
Transient Cyber Asset									Modify Alarm Settings		Manipulation of View
Wireless Compromise									Rootkit		Theft of Operational Information
									Service Stop		
									System Firmware		



Figure 24: MITRE Matrix filtered for ICS domain

As we can see in the figure above, this is the MITRE Matrix filtered for an ICS environment, and over the whole tactics the most representative ones for the ODIN Networked Components have been highlighted. The color code is based on the score provided to each tactic, depending on its criticality, and it varies from the lowest score of value 1 (green) to the highest score value of 3 (red).

Additionally, to this work of modelling with MITRE, an adaptation of the MaGMA Use Case Framework to analyze the potential threats over an industrial network has been analyzed. This analysis' results are included in ANNEX A.

3.4.3. Cybersecurity module implementation

This section describes how the cybersecurity architecture presented in section 3.3.3 is implemented and integrated in the prototype environment.

The following figure presents the main elements of the ODIN Cybersecurity system, that are a Security Information and Event Management (SIEM) system, a Security Orchestration, Automation and Response (SOAR) system and a final step of Incident Research and Resolution.

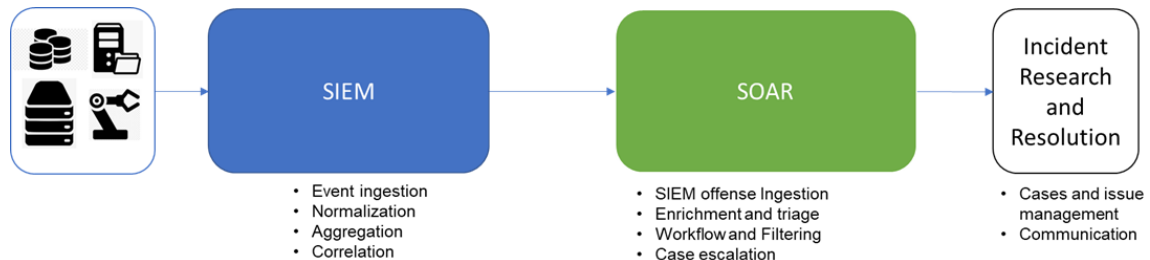


Figure 25: ODIN Cybersecurity System general implementation

SIEM:

Security information and event management (SIEM) technology supports threat detection, compliance and security incident management through the collection and analysis (both near real time and historical) of security events, as well as a wide variety of other event and contextual data sources. The core capabilities are a broad scope of log event collection and management and the ability to analyze log events and other data across disparate sources but also operational capabilities (such as incident management, dashboards and reporting).

Combining security information management (SIM) and security event management (SEM), security information and event management (SIEM) offers real-time monitoring and analysis of events as well as tracking and logging of security data for compliance or auditing purposes.

SIEM is a security solution that helps organizations recognize potential security threats and vulnerabilities before they have a chance to disrupt business operations. It surfaces user behavior anomalies and uses artificial intelligence to automate many of the manual processes associated with threat detection and incident response. This has become a staple in modern-day Security Operation Centres (SOCs) for security and compliance management use cases.

As part of a SIEM component, a SIEM agent helps to normalize and provide different actions. In ODIN, a cybersecurity agent will perform functions of an endpoint detection and response system, monitoring and collecting activity from end points that could indicate a threat. Security agent runs at a host-level, combining anomaly and signature-based technologies to detect intrusions or software misuse.

The features that SIEM agent proposed can provide among others are:

- Log collector
- Command execution
- File integrity monitoring
- Malware detection
- Container security monitoring

SOAR:

Security Orchestration Automation and Response (SOAR) is a stack of compatible software programs that enables an organization to collect data about security threats and respond to security events without human assistance. The goal of using a SOAR platform is to improve the efficiency of physical and digital security operations.

Orchestration

Connects and integrates disparate internal and external tools via built-in or custom integrations and application programming interfaces (APIs). Connected systems may include vulnerability scanners, endpoint protection products, end-user behavior analytics, firewalls, intrusion detection and Intrusion Prevention Systems (IDS/IPS), security information and event management (SIEM) platforms, as well as external threat intelligence feeds.

With all the data gathered comes a better chance at detecting threats, along with more thorough context and improved collaboration. However, the trade-off is more alerts and more data to ingest and analyze. Security automation takes action where security orchestration consolidates data to initiate response functions.

Automation

Fed by the data and alerts collected from security orchestration, it ingests and analyses data and creates repeated, automated processes to replace manual processes. Tasks previously performed by analysts, such as vulnerability scanning, log analysis, ticket checking and auditing capabilities, can be standardized and automatically executed by SOAR platforms. Using artificial intelligence (AI) and machine learning to decipher and adapt insights from analysts, SOAR automation can make recommendations and automate future responses. Alternately, automation can elevate threats if human intervention is needed.

In case that a malicious URL is found in an employee email and identified during a scan, a playbook can be instituted that blocks the email, alerts the employee of the potential phishing attempt and blocklists the IP address of the sender. SOAR tools can also trigger follow-up investigative actions by security teams if necessary. In terms of the phishing example, follow-up actions could include searching other employee inboxes for similar emails and blocking them and their IP addresses, if found.

Response

Offers a single view for analysts into the planning, managing, monitoring and reporting of actions carried out once a threat is detected. It also includes post-incident response activities, such as case management, reporting and threat intelligence sharing.

Incident research and Resolution:

This step will elevate case and issue management and will manage the notification communications. It will be performed through Security Operation Centres (SOC). SOC leverage a number of tools to detect, thwart and deal with security attacks. One of the key challenges of SOC is to quickly integrate security tools and operational activities.

The following figure presents a more detailed overview of the elements of the ODIN Cybersecurity system.

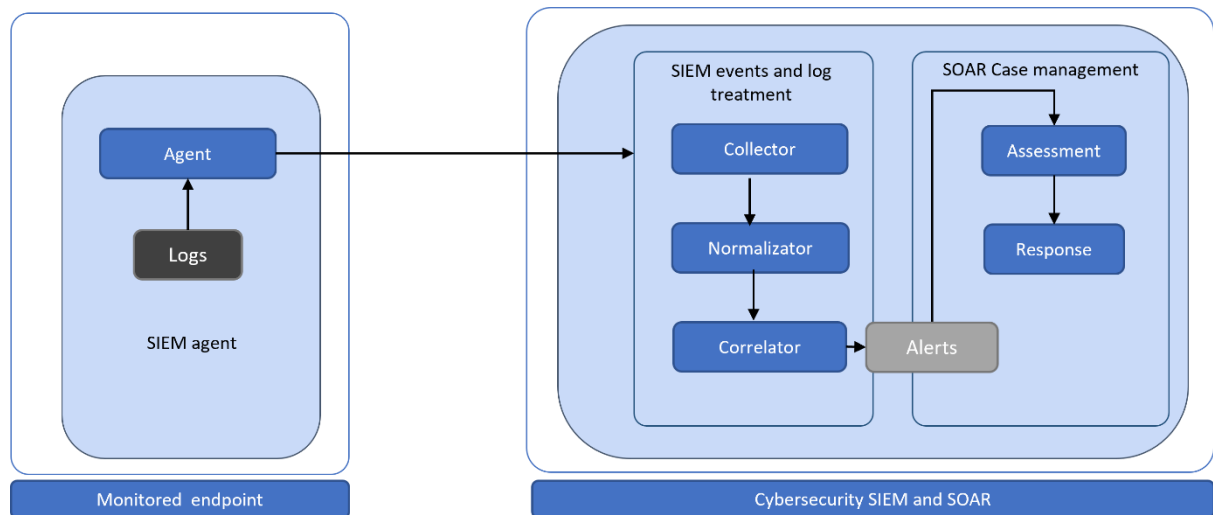


Figure 26: ODIN Cybersecurity System detailed implementation

On the one hand, there is a SIEM Agent that detect and collects the security events. In the case of the figure above the events are gathered from the logs of a monitored end-point but some other agents may also provide security events gathered sniffing the network traffic, for instance.

Then, the SIEM Agent send the events to the SIEM server, where this information is collected, normalized and correlated, so that security alerts are raised based on their criticality.

These alerts are then sent to the SOAR system, where they are further investigated to allow the appropriate case management and reactive response.

Regarding the specific description of the implementation, hereafter it is described the components of the Cybersecurity system, with the aim of a continuous incident detection and response with a SOC approach.

- Incident detection:
 - SIEM: Wazuh and Wazuh Agent / syslog / rsyslog.
 - Elasticserahc.
 - Kibana.
- Incident Response
 - SOAR:
 - Shuffle.
 - The Hive.

3.4.4. Initial integration and testing

The following figure describes the sequence to be followed for the integration between the Cybersecurity module and the OpenFlow.

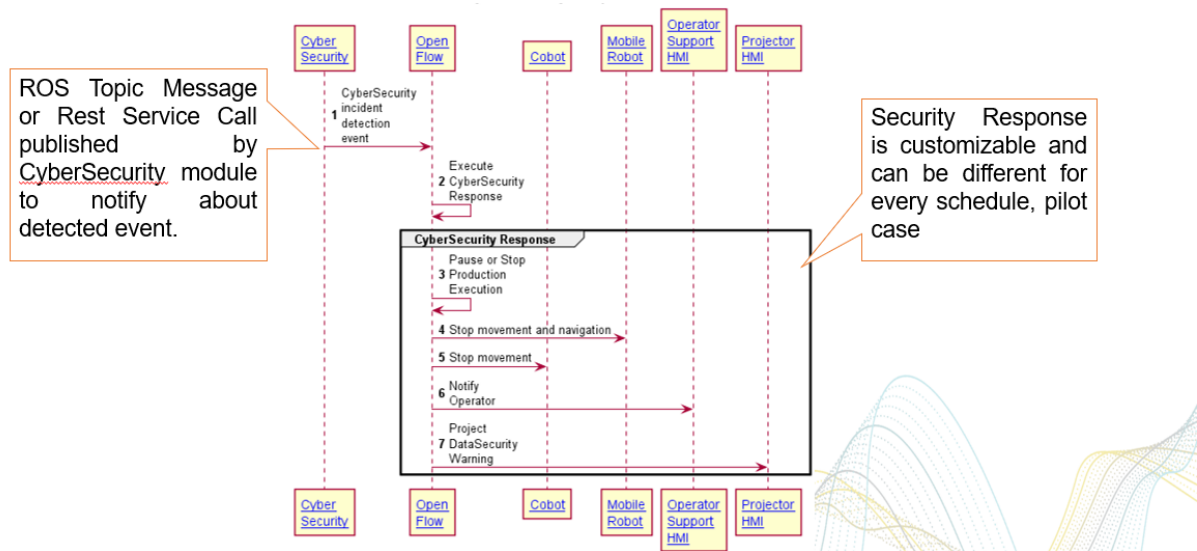


Figure 27: ODIN Cybersecurity System detailed implementation

Some attack scenario examples are summarized in the following list:

- Brute force Authentication against OpenFlow web interface.
- Exploit vulnerabilities in ROS and launch remote code executions from a privileged ROS endpoint compromising completely the computational graph.
- Robot Vulnerability Database for ROS1 implementation and according to ROS Noetic version.

In addition, focus is given on the presentation of the full chain of detection and response by the ODIN Cybersecurity System for the Brute Force Attack.

1. SIEM: The event is detected Wazuh.

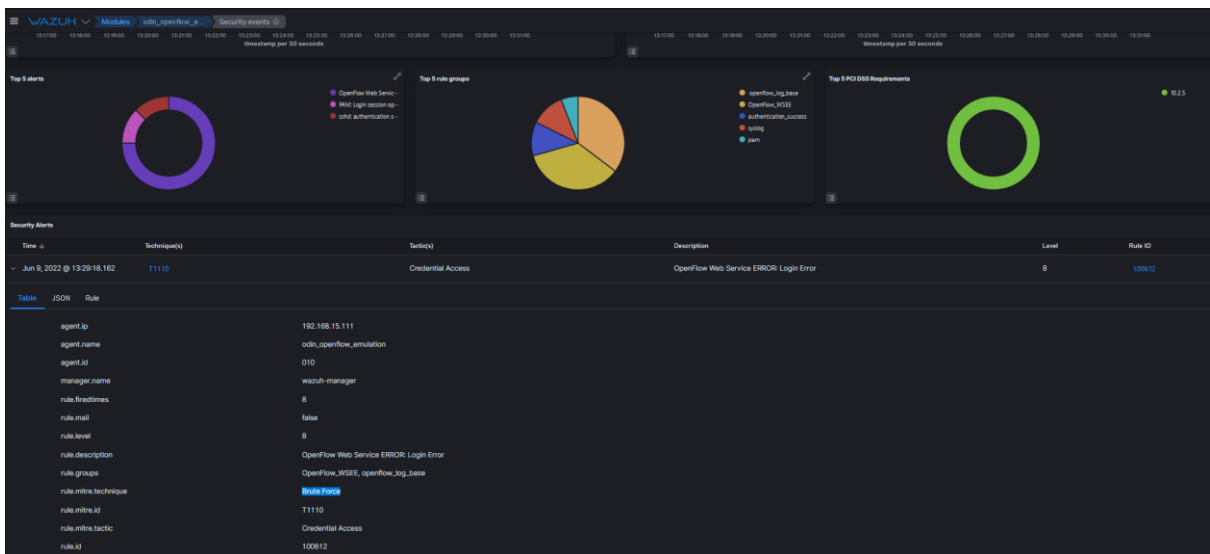


Figure 28: Event detected in the SIEM

2. SOAR - Shuffle: There is a workflow that ingests events from the SIEM and performs an automated case management in Shuffle. The workflow steps are the following:
 - 2.1.SIEM phase: Webhook connected to the SIEM that receives the events and transfers them to the Node step.
 - 2.2.Node: It will get the data from the SIEM and will parse the information for the Tool step (data driven).
 - 2.3.Tool: Tool for data driven process that will prepare the data in JSON format.
 - 2.4.Create alert (The Hive): Here there is a condition that states that if the alert criticality score is higher that a predefined threshold, the alert will be created in The Hive.

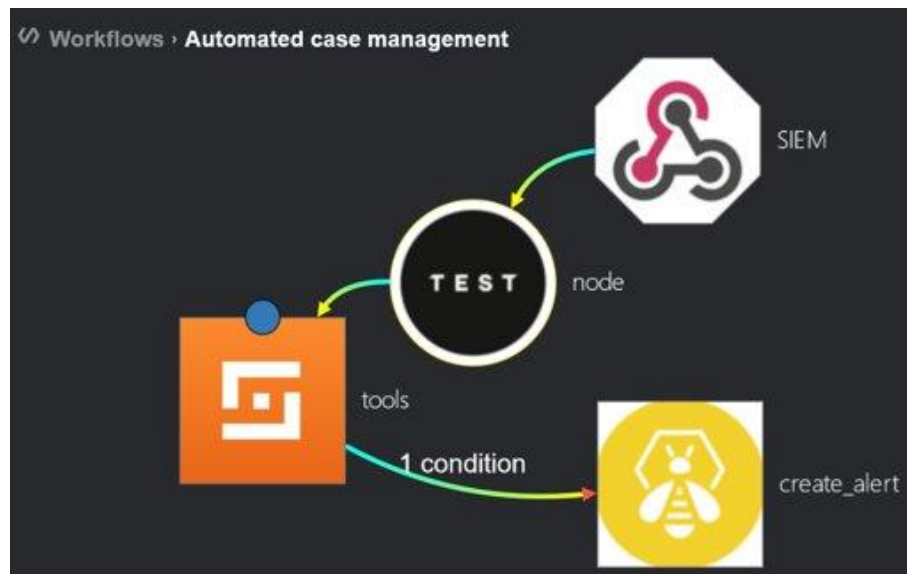


Figure 29: Automated case management in Shuffle

3. SOAR - The Hive: The alert is forwarded to The Hive automatically:

List of alerts (4 of 4)

No event selected - Quick Filters - Sort by - Custom Fields Stats Filters 15 per page

Filters
+ Add a filter

Severity	Read	Title	# Case	Type	Source	Reference	Observables	Dates	O	C	U
<input type="checkbox"/>	M	Unread OpenFlow Web Service ERROR: Login Error	None	Alert_Incident	SIEM-1654778790.4966602	-2022-06-09T12:46:30.303+0000WAZUH Alert	0	O. 06/09/22 14:47 C. 06/09/22 14:47			
ros OpenFlow Web Service ERROR: Login Error 100612 192.168.15.111 Jun 9 12:46:29 ros docker/odin_openflow_emulation(657221): 12:46:29.373 [http-nio-8082-exec-2] ERROR eu.robotics.station_controller.ui.services.web1 - Login Error: Unknown user: [hack]											
None											
<input type="checkbox"/>	M	Unread OpenFlow Web Service ERROR: Login Error	None	Alert_Incident	SIEM-1654778792.4967439	-2022-06-09T12:46:32.264+0000WAZUH Alert	0	O. 06/09/22 14:47 C. 06/09/22 14:47			
ros OpenFlow Web Service ERROR: Login Error Jun 9 12:46:30 ros docker/odin_openflow_emulation(657221): 12:46:30.761 [http-nio-8082-exec-2] ERROR eu.robotics.station_controller.ui.services.web1 - Login Error: Unknown user: [hack] 100612 192.168.15.111											
None											
<input type="checkbox"/>	M	Unread OpenFlow Web Service ERROR: Login Error	None	Alert_Incident	SIEM-1654778792.4967035	-2022-06-09T12:46:30.259+0000WAZUH Alert	0	O. 06/09/22 14:47 C. 06/09/22 14:47			
ros OpenFlow Web Service ERROR: Login Error Jun 9 12:46:30 ros docker/odin_openflow_emulation(657221): 12:46:30.761 [http-nio-8082-exec-2] ERROR eu.robotics.station_controller.ui.services.web1 - Login Error: Unknown user: [hack] 100612 192.168.15.111											
None											
<input type="checkbox"/>	M	Unread OpenFlow Web Service ERROR: Login Error	None	Alert_Incident	SIEM-1654778790.4966197	-2022-06-09T12:46:30.259+0000WAZUH Alert	0	O. 06/09/22 14:46 C. 06/09/22 14:46			
ros OpenFlow Web Service ERROR: Login Error 100612 192.168.15.111 Jun 9 12:46:29 ros docker/odin_openflow_emulation(657221): 12:46:29.373 [http-nio-8082-exec-2] ERROR eu.robotics.station_controller.ui.services.web1 - Login Error: Unknown user: [hack]											
None											

Figure 30: Alert management in The Hive

- SOAR – The Hive: The Hive is able to create a case based on the alert, with the aim of further research and respond to the incident through other SOC capabilities.

The screenshot displays the The Hive interface. At the top, there is a navigation bar with 'TheHive' logo, 'New Case', 'My tasks', 'Waiting tasks', 'Alerts', 'Dashboards', and 'Search'. The user is logged in as 'Casold' and the organization is 'ODIN/odin_shuffle'.

The main area shows a 'List of cases (1 of 1)'. A filter section is visible with 'No case selected', 'Quick Filters', and 'Sort by'. Below this is a table of cases:

Status	#	Number	Title	Severity	Details	Assignee	Dates	S.	U.	C.
Open	3	#10	OpenFlow Web Service ERROR: Login Error	High	Tasks	0	S. 06/09/22	14:51	06/09/22	14:51

The detailed view on the right shows the alert information for the selected case:

- Alert:** OpenFlow Web Service ERROR: Login Error
- Description:** Jun 9 12:46:29 ros docker/odin_openflow_emulation[657221]: 12:46:29.373 [http-nio-8082-exec-20] ERROR eu.robotics.station_controller.ui.services.web.i - Login Error: Unknown user: [hack]
- SIEM-1654778790.4966602] OpenFlow Web Service ERROR: Login Error**
- Severity:** High
- Tags:** ros, OpenFlow Web Service ERROR: Login Error, 100612, 192.168.15.111, Jun 9 12:46:29 ros docker/odin_openflow_emulation[657221]: 12:46:29.373 [http-nio-8082-exec-20] ERROR eu.robotics.station_controller.ui.services.web.i - Login Error: Unknown user: [hack]
- Tip:** TLP:AMBER
- Pap:** 2
- Read:** false
- Custom Fields:** []
- Observable Count:** 0
- Extra Data:** []

Figure 31: Alert escalation to case in The Hive

4. CONCLUSIONS

The main aim of the deliverable D4.1 is the presentation of the initial prototype of:

- OpenFlow module, which is responsible to integrate, orchestrate, manage and coordinate production resources to execute manufacturing schedules.
- Cyber Security module, which uses models and use cases, monitors and provides detection and response capabilities on the deployed Network Component.

The work performed at this stage of the project has been focused on demonstrating the functionality of the prototypes. Then in M36, the final version will be provided, including further work for extending the features of the prototypes, continuing the integration with use cases and completing the validation phase.

The OpenFlow initial prototype has been presented in this document. It is a functional, initial prototype that has also demonstrated integrated functionality in the scope of the small-scale pilot case developments of T2.6. A positive outcome was the first initial prototype of most features has been implemented until M18 and this not only facilitated the integrated execution and testing of the small-scale Pilot Cases but also provided valuable information and feedback that will help the future steps of the development of the OpenFlow modules.

The next steps of the OpenFlow development will closely follow and support the work done in other work packages and modules aiming to release more internal versions, in order to support the Pilot Case development and integration activities. At the same time efforts will be focused to extend and implement all OpenFlow required features and particularly the higher-level features such as the monitoring of the other modules status and the re-scheduling of the production execution.

In addition, the Cybersecurity initial prototype has been presented in this document. It has demonstrated its three main features of threat modelling, detection and response over an emulated scenario of the ODIN Networked component and through an example of the Brute Force attack technique.

The next steps will be focused on completing the threat model adapted for the ODIN scope and including all the relevant tactics. On the other hand, the integration of the response actions with advanced SOC management functionalities will be investigated. Finally, the validation phase will be performed as well.

Overall, the initial prototype of the Network Component has been presented, including a functional initial prototype of the OpenFlow and Cybersecurity modules.

WP4 will continue working on the development of the ODIN Network Component in close collaboration with the other WPs and at the same time WP4 will work on the deployment and testing at pre-industrial scale that takes place in T4.3. The plan is to have a preliminary deployment and testing completed by M24, that will be documented in D4.2.

5. GLOSSARY

AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
CRUD	Create, Reade, Update, Delete
C&C	Command and Control
C2	Command and Control
DB	Database
DDD	Domain Driven Design
ERP	Enterprise Resource Planning
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
IP	Internet Protocol
IPS	Intrusion Prevention System
ISA	Industry Standard Architecture
IT	Information Technology
HMI	Human Machine Interface
HRC	Human Robot Collaboration
KR	Knowledge Repository
MES	Manufacturing Execution Systems
OSINT	Open-Source Intelligence
OT	Operational Technology
PLM	Product Lifecycle Management
ROS	Robot Operating System
SCADA	Supervisory Control and Data Acquisition
SOA	Service Oriented Architecture
SOAR	Security Orchestration, Automation and Response
SIEM	Security Information and Event Management
SOC	Security Operation Centre
UI	User Interface
URL	Uniform Resource Locator

6. REFERENCES

1. Chryssolouris, G., Manufacturing Systems: Theory and Practice, 2nd Edition, Springer-Verlag, New York, New York, (2006)
2. G. Michalos, S. Makris, N. Papakostas, D. Mourtzis, G. Chryssolouris, "Automotive assembly technologies review: challenges and outlook for a flexible and adaptive approach", CIRP Journal of Manufacturing Science and Technology, Volume 2, Issue 2, pp. 81-91 (2010)
3. N. Kousi, S. Koukas, G. Michalos, S. Makris: "Scheduling of smart intra – factory material supply operations using mobile robots", International Journal of Production Research, Volume 57, Issue 3, pg. 801-814, (2018)
4. N. Kousi, S. Koukas, G. Michalos, S. Makris, G. Chryssolouris, "Service oriented architecture for dynamic scheduling of mobile robots for material supply", CIRPe2016 , Procedia CIRP, 5th CIRP Global Web Conference-Research and Innovation for Future Production [Volume 55, pp. 18-22](#) (2016)
5. S. Papanastasiou, N. Kousi, P. Karagiannis, C. Gkournelos, A. Papavasileiou, K. Dimoulas, K. Baris, S. Koukas, G. Michalos, S. Makris, "Towards seamless human robot collaboration: integrating multimodal interaction", The International Journal of Advanced Manufacturing Technology, [Volume 105, pg. 3881-3897](#), (2019)
6. G. Michalos, N. Kousi, P. Karagiannis, C. Gkournelos, K. Dimoulas, S. Koukas, P. Mparis, A. Papavasiliou, S. Makris, "Seamless human robot collaborative assembly – An automotive case study", Mechatronics, [Volume 55, pg 194-211](#), (2018)
7. S. Makris, P. Karagiannis, S. Koukas, A. S. Matthaiakis, "Augmented reality system for operator support in human–robot collaborative assembly", CIRP Annals - Manufacturing Technology, [Volume 65, Issue 1, pp. 61-64](#), (2016)
8. S. Koukas, N. Kousi, S. Aivaliotis, G. Michalos, R. Bröchler, S. Makris, "ODIN architecture enabling reconfigurable human – robot based production lines", Procedia CIRP, [Volume 107, pg 1403-1408](#), (2022)
9. Official Docker site, Docker, <https://www.docker.com/> accessed online 2021-09
10. OWASP, [attack surface analysis: https://cheatsheetseries.owasp.org/cheatsheets/Attack_Surface_Analysis_Cheat_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Attack_Surface_Analysis_Cheat_Sheet.html)
11. "Domain Driven Design, Definitions and Pattern Summaries", E. Evans, (2015). https://domainlanguage.com/wp-content/uploads/2016/05/DDD_Reference_2015-03.pdf accessed online 2022-05.
12. OWASP, threat modelling: https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html
13. MITRE ATT&CK: <https://attack.mitre.org/>
14. ISA99: Security for Industrial Automation and Control Systems: <http://www.enerxis.com/en/industry-standards/isa99.php>
15. Cyber Kill Chains applied to ICS: <https://www.incibe-cert.es/en/blog/cyber-kill-chain-applied-ics>.
16. MaGMA: <https://www.betaalvereniging.nl/en/safety/magma/>
17. International Electrotechnical Commission (IEC), Industrial communication networks - network and system security, Geneva, 2018.

7. ANNEX A: MaGMA ADAPTATION TO INDUSTRIAL NETWORKS

This report proposes the adaptation of the MaGMA Use Case Framework to analyze the potential threats over an industrial network.

The analyzed scenario is based on a simple assembly line operated by a ROS-powered robot. The following reference documentation was used to build the scenario:

- NIST Special Publication 800-82, Guide to Industrial Control Systems (ICS) Security [1].
- ISA/IEC 62443 family of standards [2].
- Red Team Robot Security White paper from Alias Robotics [3].
- MaGMA Use Case Framework [4].

In addition to the present document, the obtained MaGMA analysis performed using the excel document of Magma UCF tool [7] adjusted on ODIN ROS based system.

7.1. Steps of the Analysis

The execution of the MaGMA Use Case Framework (hereafter, called MaGMA) requires the definition of a process of analysis:

1. **Define the scope:** Describe the scenario that is going to be analyzed.
2. **Identify relevant assets and potential entry points:** Define which elements in the scope are going to be inside the analysis, and which potential entries can be used to compromise them.
3. **Analyze Potential Drivers and References in MaGMA:** Specify which detection technologies are implemented in the scenario, which are the potential threat actors, or any compliance driver that has to be fulfilled.
4. **Generation of L1 use cases:** Generation of L1 use cases using the extended cyber kill chain defined in the MaGMA tool.
5. **Generation of L2 use cases:** Each L1 use case is expanded using the list provided in the MaGMA tool to generate the corresponding L2 use cases.
6. **Generation of L3 use cases (ATT&CK Matrix):** Finally, L3 use cases for each L2 use case are generated using the ATT&CK Matrix as a reference.

The next figure provides a flow diagram of the steps that should be followed to perform an analysis of a given scenario using the MaGMA Use Case Framework.

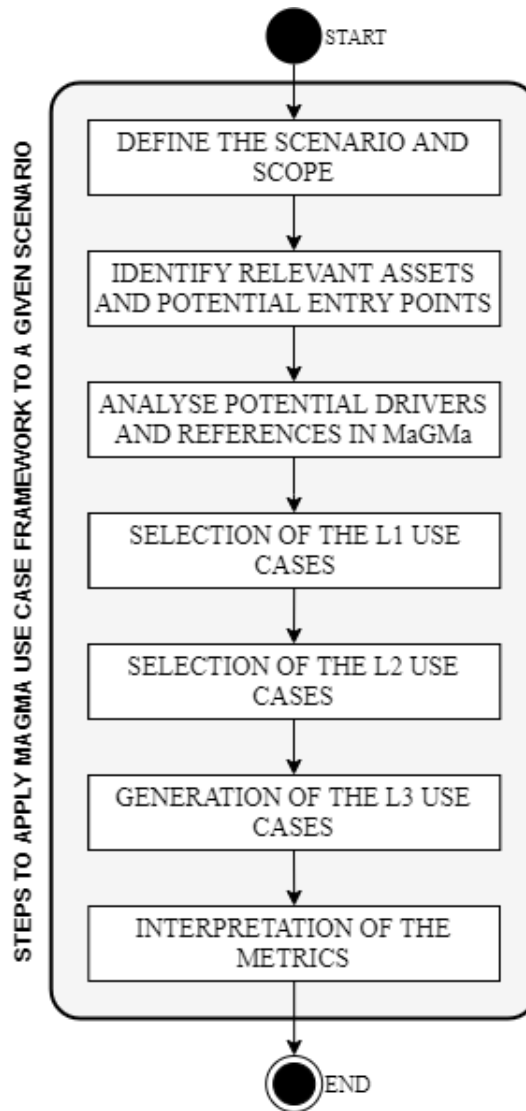


Figure 32: Steps of MaGMA use case framework for a given scenario

In the following sections, each one of these steps is further explained, and applied to a use case that serves as an example.

IMPORTANT NOTE: It is important to notice that the formulas provided in the MaGMA tool do not work properly, especially when rows are filtered (e.g., alphabetically). To correct them, we have to go to the “L2 UC” page on the document. For every formula defined between column “H” and column “L”, we have to delete every 'L2 UC!' that appears in the formulas. An example of this process is presented in Figure 33. An example of how the formula should look like is shown in Figure 34.

	H	I	J	K	L
	Avg Effectiveness	Avg Implementation	Avg Coverage	Avg Weight	Avg Potential
	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%

Figure 33: Relative reference in the formulas of the MaGMA tool

Relative reference in the formulas of the MaGMA tool must be deleted in order for the tool to work properly. This has to be done for every formula from column “H” to column “L”.

	H	I	J	K	L
	Avg Effectiveness	Avg Implementation	Avg Coverage	Avg Weight	Avg Potential
	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%

Figure 34: Modified formulas in the MaGMA tool

7.2. Step 1 - Definition of the Scope

The analyzed scenario is a simplified version of an assembly line operated by a ROS-powered robot, shown in Figure 35 [3]. The network generalizes the main elements that can usually be found in a similar environment, together with the subnetworks recommended by the NIST SP 800-82 special publication [1].

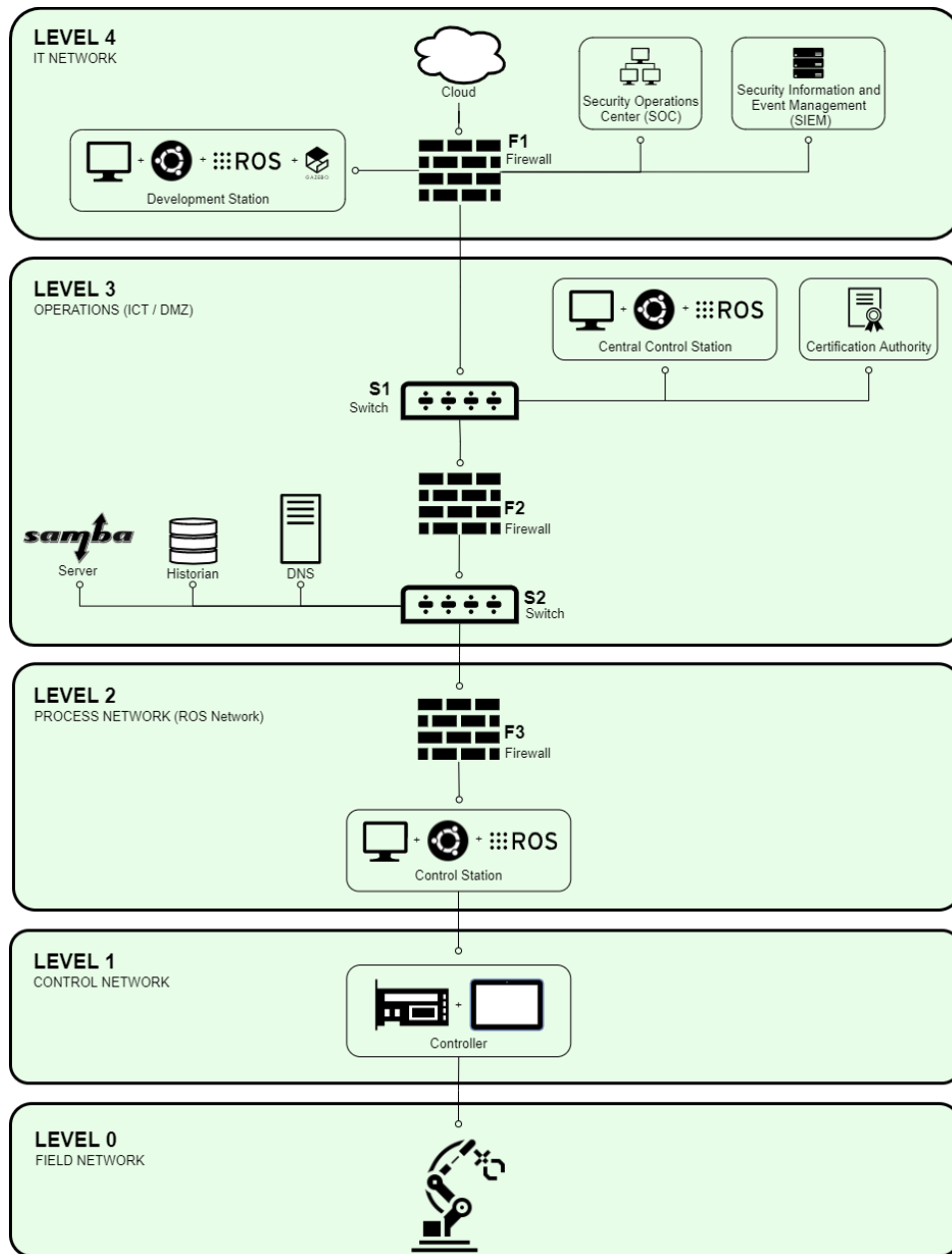


Figure 35: Use case architecture diagram

This synthetic scenario presents a network segmented in 5 levels with segregation implemented following recommendations in NIST SP 800-82 and IEC 62443 family of standards. There are 6 identical robots from Universal Robots [6] presenting a variety of networking setups and security measures, each connected to their controller. ROS and ROS-Industrial package live in Levels 2, 3 and 4.

The ruleset that controls the communication between boundaries is defined as follows:

- The first firewall F1 blocks arbitrary packages from the Internet to enter the IT Network (Level 4). Only selected traffic should be allowed from proceeding to the enterprise network.

- The second firewall, F2, blocks packages in the IT Network (Level 4) from going to the OT networks (Level 2 and below).
- The third firewall F3 only allows permitted traffic from the DMZ (Level 3) to the OT Networks (Level 2 and below). Connections between Level 3 and Level 2 are permitted only when they are initiated by the endpoint in Level 2.

It is important to remark that both external and insider attacks must be considered to make the analysis more comprehensive.

7.3. Step 2 - Identification of the relevant assets and Entry Points

In this step, each one of the assets that will be part of the analysis is defined and characterized. Moreover, for each asset, the potential entry points will be also defined. Table 8 provides a summary of the key elements in the network.

7.3.1. Central Control Station

It is a Linux-based central control station which command other ROS-enabled endpoints (such as the ROS drivers enabled on each sub-control station):

- Ubuntu Bionic (18.04 LTS).
- ROS Noetic Ninjemys 1.15.14.
- ROS-Industrial packages, communicating with the robot controller via a local area network.
- Security measures applied follow the recommendations of Canonical's report (Canonical, 2020) on how to secure ROS robotics platforms in Ubuntu Bionic 18.04 Linux distribution. No wireless communications are assumed to be enabled.
- The central control station is assumed unique in the networking setup and wherein the ROS Master process will be running (in other words, all other ROS-enabled machines will be acting as slaves).

7.3.2. Certification Authority

A certificate authority or certification authority (referred as CA in both cases) is an entity that issues digital certificates. In the context of the use case, the CA is represented by either an individual machine or a process running in the Central Control Station that issues digital certificates which certify the ownership of a public key by the named subject (another entity in the use case) of the certificate. This allows others (relying parties) to rely upon signatures or on assertions made about the private key that corresponds to the certified public key. The CA acts as a trusted third party—trusted both by the subject (owner) of the certificate and by the party relying upon the certificate. The format of these certificates is specified by standards (generally the X.509). The CA could be either continuously operating and serving or be switched off by default and get enabled only when new certificates need to be issued.

7.3.3. Historian Database

A historian is a software service that accumulates time-stamped data, events, and alarms in a database which can be queried or used to populate graphic trends in the HMI.

7.3.4. Control Station

It is a Linux-based control station which operates the robot controller (and coherently, the robot mechanics):

- Ubuntu Bionic (18.04 LTS).
- ROS Noetic Ninjemys 1.15.14.
- ROS Industrial drivers for Universal Robots, communicating with the robot controller via a local area network.
- No wireless connectivity is assumed.
- Security measures applied follow the recommendations of Canonical's report [5] on how to secure ROS robotics platforms in Ubuntu Bionic 18.04 Linux distribution. No wireless communications are assumed to be enabled.

7.3.5. Controller

The robot controller is accessible locally via physical means (e.g., USB ports or Ethernet ports) or its local network connections.

The controller includes by default no security measures enabled. Since the initial use case was based on the preliminary white goods scenario, each controller is assumed to run firmware version 3.13.0 from Universal Robots.

Table 8: Summary of the assets in the network of the use case and their properties

ASSET	HARDWARE	SW	ENTRY POINTS	SECURITY MEASURES
Central Control Station	<ul style="list-style-type: none"> - Industrial-grade PC - 4 cores CPU - 4096 MB RAM 	<ul style="list-style-type: none"> - Ubuntu Bionic (18.04 LTS) - ROS Noetic Ninjemys (1.15.14) - ur_modern_driver - Universal_Robots_ROS_Driver 	<ul style="list-style-type: none"> - Physical access (local area network interfaces, storage devices, etc.) - Local area network 	Beyond the defaults, no particular security measures are applied into the control stations.
Control Station	<ul style="list-style-type: none"> - Industrial-grade PC - 4 cores CPU - 4096 MB RAM 	<ul style="list-style-type: none"> - Ubuntu Bionic (18.04 LTS) - ROS Noetic Ninjemys (1.15.14) - ur_modern_driver - Universal_Robots_ROS_Driver 	<ul style="list-style-type: none"> - Physical access (local area network interfaces, storage devices, etc.) - Local area network 	Beyond the defaults, no particular security measures are applied into the control stations.
Controller	Universal Robots controller CB3.1	-N/A	<ul style="list-style-type: none"> - Teach pendant - Ethernet port - USB port (in the teach pendant) - Local area network 	Beyond the defaults, no particular security measures are applied into the control stations.

7.4. step 3 - Analyse Potential Drivers and References in MaGMa

Before start building the use cases, MaGMa requires to fulfil the Drivers and References in the tool:

- Business Drivers.
- Compliance Drivers.
- Threat Actors.
- Detection Technologies.
- Log Sources.
- Scope.

It is worth noticing that business drivers, as they are defined by MaGMa, are not always relevant for the analysis. This is happening because they give a high-level view and most of the time the analysis is focused on low-level systems, and not on a whole organization.

For those cases, it is proposed to use the primary and secondary properties of security instead:

- Confidentiality.
- Integrity.
- Availability.
- Accountability.
- Authenticity.
- Non-Repudiation.

This method complements the information generated by MaGMa for each use case, in order to see which dimension is the most affected at the end of the analysis.

7.4.1. Potential Drivers and References in the Ros-based Scenario

The default settings in the MaGMa tool for all business drivers, detection technologies, log sources, and scope are being used. Nevertheless, new compliance drivers and more precisely, new external regulators have been defined as follows:

- ISA/IEC 62443-4-1.
- NIST Special Publication 800-82.
- ISO 27001.
- ISO 27005.

Additionally, existing threat actors have been modified assuming that:

- There are no know open conflicts with any third country.
- No terrorist organization is interested in our production process.
- Hackavists, Cyber Vandals, and Script Kiddies do not target organizations whose main activity is focused on automated manufacturing. This means that our scenario is out of scope for them.
- There is no social engineering implied.

This means that the actors that which might target the ODIN network are:

- Professional criminals.
- Internal actors.
- Private organizations.

- Multiple actors.

When more than two actors apply to any given use case, “multiple actors” will be used. A new actor “None” has been defined to indicate that the corresponding use case does not apply to the scenario that is being analyzed.

Finally, business drivers, and external policies where not used in this analysis.

IMPORTANT NOTE: These data are filled in the “Drivers” and “References” pages of the MaGMA Tool. In these pages, more rows can be added freely.

7.5. Step 4 – Generation of the L1 Use Cases

In this step, the initial threat categories proposed in the L1 use case level in MaGMA, to detect possible missing categories are reviewed. We can move to the next step in case that no missing threat category is detected. An overview of the L1 use cases is presented in Figure 36.

Strategic Overview						
Threat category	L1 Use Case Identifier	Use Case Name	Purpose	Stakeholders	#L2 UC related	#L3 UC related
Cyber killchain	RE	Reconnaissance			7	0
	N/A	Weaponization				
	DE	Delivery			7	0
	EX	Exploitation			5	16
	IN	Installation			2	0
	CC	Command & Control			2	16
	AO	Actions on Objectives			16	126
Other threats	FE	Fraud / Extortion			2	0
	DD	(D)DoS			3	0
	PH	Physical Access Compromise			5	0
	BL	Blacklisting			2	0
	SD	Sabotage / Destruction			3	0
	PV	Policy Violations			8	11

Figure 36: Threat categories proposed in MaGMA to serve as an overview of the use cases

These threats categories proposed by MaGMa aim to serve as a classification mechanism at high level for all the use cases. This classification gives an overview of the current status of the existing use cases. For this reason, the MaGMa version of the cyber kill chain is used in this report.

It is worth noticing that there exist many modifications of the cyber kill chain model originally proposed by Lockheed Martin Corporation. These new proposals modify the original model, adapting it to meet the features of each particular scenario. An example of this can be found in the white paper published by Alias Robotics, where they adapted the initial model to reflect the characteristics of a robotic environment [3].

The main difference with the cyber kill chain used in MaGMa is that the one proposed by Alias Robotics consider both external but also insider attacker and adapts the steps consequently. Nevertheless, this feature is easy to be integrated in MaGMa during the analysis.

7.5.1. L1 Use Cases for the ROS-based Scenario

In the scenario presented in this document, no additional threat categories were needed, so the analysis was performed using the ones proposed by MaGMa by default.

IMPORTANT NOTE: In this step, the only cells that can be modified are those that belong to columns “E” and “F”. The remaining cells in the L1 UC page are automatically filled. New rows should be added below the existing row 16 in case that more threat categories are needed. The rest of the document should be manually updated to reflect this change.

7.6. Step 5 – Generation of the L2 Use Cases

In this step, each individual L2 use case should be checked to find whether it applies to the scenario that is being analyzed or not. In addition, information related to actors, business drivers, internal policies, and external regulators could be added if it is required.

MaGMa already provides a comprehensive list of 62 L2 use cases, associated to the initial L1 use cases, as is shown in Figure 37.

L1 Use Case Name	L1 Use Case Identifier	L2 Use Case Identifier	Use Case Name	Use Case Description
Exploitation	EX	EX-IDS	Network intrusion attempt	Any direct intrusion attempt into the network, for example by using exploits against systems in the network. Intrusion attempts can be directed at mail servers, web servers or any other internet-facing servers
Exploitation	EX	EX-NET	Network-based attack	Network-based attacks aimed at manipulating the information transmitted by the user across the network. This includes session hijacking and MITM attacks
Fraud / Extortion	FE	FE-CEO	CEO Fraud	Attempts by criminals to impersonate the organization's CEO or other board-level executive and to have the victim transfer large amounts of money. Usually carried out by sending spoofed email or by business email compromise
Fraud / Extortion	FE	FE-EXT	Extortion attempts	Any attempts by attackers to gain financial compensation from the organization by not taking action. Such action may include DDoS attacks or sensitive information disclosure
Installation	IN	IN-MAL	Installation of malware	Attempted installation of malware onto an end-point
Installation	IN	IN-RAT	Installation of Remote Access Tool or similar backdoor	Any attempt (both successful and failed) to install a Remote Access Tool (RAT) onto an end-point
Physical Access Compromise	PH	PH-AAT	Anomalous access token usage	Any anomalous usage of access tokens, such as entry badges
Physical Access Compromise	PH	PH-DCE	Illegal entry into datacenter	Physical access compromise by an attacker into the data center
Physical Access Compromise	PH	PH-MOB	Mobile device lost or stolen	Mobile device, such as a tablet or a phone lost or stolen
Physical Access Compromise	PH	PH-NAC	Physical network access compromise	This is usually placement of a physical rogue device in the network. Such a device could be a wireless access point (WAP) or any other device, including Raspberry pi
Physical Access Compromise	PH	PH-OFE	Illegal entry into office space	Physical access compromise by an attacker into the office space
Policy Violations	PV	PV-ACC	Unauthorized use of high-privileged account	Any unauthorized use of accounts with high privileges, such as root accounts on Linux / UNIX and administrator accounts on Windows

Figure 37: Examples of the L2 use cases provided by MaGMA

7.7. L2 Use Cases for the ROS-based Scenario

The generation of the corresponding L2 use cases for the ROS-based scenario consists of analyzing each one of the 62 L2 UC proposed by MaGMA. For each occasion, it needs to be decided whether it applies to the investigated ROS-based scenario or not.

It is possible to detect a potential L2 UC that is not in the proposed list by MaGMA. In that case, new row at the bottom to integrate those in the analysis can be added.

In addition, when a L2 UC applies, it is required to fulfil its business drivers, internal policies, and internal regulators when necessary. This means that not every L2 UC might have a business driver associated, nor an internal policy, nor an internal regulator associated.

Figure 38 shows the extracted L2 UC generated for the ROS-based scenario.

L1 Use Case Name	L1 Use Case Identifier	L2 Use Case Identifier	Use Case Name	Use Case Description	Actors	#L3 UC relate	Avg Effectivene	Avg Implementati	Avg Coverag	Avg Weigt	Avg Potent	Business Driver	Internal Policy	External regulator	External regulator
Delivery	DE	DE-PHI	Delivery of phishing mail	(Attempted) delivery of phishing mails to recipients within the organization, including CEO fraud emails	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable
Delivery	DE	DE-PHY	Physical malware delivery	Delivery of malware via physical means, such as a USB drive	Internal Actors	1	0%	0%	0%	0%	0%	Not applicable	Not Applicable	NIST 800-82	NIST 800-82
Delivery	DE	DE-RAT	Delivery of Remote Access Tool	(Attempted) delivery of a remote access tool to a user within the organization	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable
Delivery	DE	DE-WEB	Web based malware delivery	Delivery of malware via malicious web pages, including exploit kits	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable
Delivery	DE	DE-WMO	Widespread malware outbreak	This is a use case that may cover other L2 malware related use cases. Reserved for use in case of multiple outbreaks within a short period, possibly across organizational domains or geographic locations	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable
Delivery	DE	DE-WRM	Worm propagation	Delivery of malware through automated propagation mechanisms. Such mechanisms are usually vulnerable protocols allowed within the organization	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable
Exploitation	EX	EX-APP	Application intrusion attempt	Attempt to exploit a service, possibly resulting in application or service crashes	Multiple Actors	1	0%	0%	0%	0%	0%	Not applicable	Not Applicable	ISO27005	Not Applicable
Exploitation	EX	EX-AWE	Application whitelisting evasion attempt	Any attempts to circumvent application whitelisting controls on end-points	Multiple Actors	15	0%	0%	0%	0%	0%	Not applicable	Not Applicable	IEC62443-4-1	NIST 800-82
Exploitation	EX	EX-BRU	Brute force exploitation attempt	Attempts to break into accounts and systems by performing a password guessing attack	Multiple Actors	1	0%	0%	0%	0%	0%	Not applicable	Not Applicable	IEC62443-4-1	Not Applicable
Exploitation	EX	EX-IDS	Network intrusion attempt	Any direct intrusion attempt into the network, for example by using exploits against systems in the network. Intrusion attempts can be directed at mail servers, web servers or any other internet-facing servers	Multiple Actors	1	0%	0%	0%	0%	0%	Not applicable	Not Applicable	ISO27005	Not Applicable
Exploitation	EX	EX-NET	Network-based attack	Network-based attacks aimed at manipulating the information transmitted by the user across the network. This includes session hijacking and MITM attacks	Multiple Actors	1	0%	0%	0%	0%	0%	Not applicable	Not Applicable	IEC62443-4-1	Not Applicable
Fraud / Extortion	FE	FE-CEO	CEO Fraud	Attempts by criminals to impersonate the organization's CEO or other board-level executive and to have the victim transfer large amounts of money. Usually carried out by sending spoofed email or by business email compromise	None	0	0%	0%	0%	0%	0%	Not applicable	Not Applicable	Not Applicable	Not Applicable

Figure 38: Example of the selected L2 use cases for the ROS-based scenario

IMPORTANT NOTE: In this step, the only cells that can be modified are those that belong to columns “F”, “M”, “N”, “O”, and “P”. The remaining cells in the L2 UC page are automatically filled. If more L2 Use Cases are needed, they should be added after row 63. If more Internal Policies, or external regulator are needed, they should be added after column “P”. The rest of the document should be manually updated to reflect this change.

7.8. Step 6 – Generation of the L3 Use Cases

This is the most critical step in all the process, so it is important to be methodical to consider all options. This is because, at this step the real use cases are developed and related to the initial system that is analyzed. In this step, for each L2 UC it is required to analyze:

- What event or events should be detected in order to fulfil the L2 UC.
- Which techniques of the ATT&CK matrix can be used by the attackers to cause such events.

In addition, the log source that is going to be used to detect that L3 UC, the detection technology and the scope should be added for each L3 UC which is completed.

Regarding to the scope, it is important to notice that this field can be used to indicate which asset of the system is affected by this L3 UC. Moreover, more than one “Scope” column can be used to even detail more the scope of each L3 UC.

Finally, for each L3 UC, it is required to specify the value of:

- **Effectiveness percentage:** This metrics indicates how effective the assigned detection mechanism is to detect the corresponding event. For example, a proxy inspecting traffic is much less effective if it is not able to inspect HTTPS traffic.
- **Implementation percentage:** This metric indicates how well a detection mechanism has been implemented. For example, the implementation level of an IDS is much lower if the ruleset is incomplete or has not been tuned.
- **Coverage percentage:** This metrics indicates the level in which this detection mechanism covers the use case. For example, a use case focused on firewall events has less coverage if not all traffic is routed through the connected firewall.

IMPORTANT NOTE: There exist multiple ATT&CK matrices provided by MITRE, categorized in three main groups: 1. Enterprise, 2. Mobile, and 3. ICS. It is worth pointing out that the enterprise matrix was used in this report.

7.8.1. L3 Use Cases for the ROS-based Scenario

To generate the L3 UC for the ROS based scenario, the potential techniques and event to each L2 UC detected were associated. Initially, all the possible L3 UC associated to each L2 UC should be considered. Nevertheless, the technical documentation or standards to support our statements, together with the example provided in MaGMA can be used.

The three associated L3 UC for the L2 UC “Account breached.” are presented in Figure 39. Although MaGMA provides most of the L2 UC in a comprehensive manner, L3 use cases have to be extracted by the analyst according to the scenario being analyzed.

An initial approach to extract L3 UCs is to find related key terms in the ATT&CK matrix. This can serve as a good starting point to find both the L3 US and the technique associated to them.

# L3 use cases per L2	L1 Use Case Name	L1 Use Case Identifier	L2 Use Case Name	L2 Use Case Identifier	Rule Identifier	Technical Use Case Name	Use Case Description	Effectiveness %	Implementation %	Coverage %	Weight (eff*impl*cvrg)	Potential (eff-weight)	Log source type	Detection Technology	Scope
3	Actions on Objectives	AO	Account breached	AO-ACC	AO-ACC-01	Detect Unauthorized Usage of Valid Accounts	https://attack.mitre.org/wiki/Technique/T1078	0%	0%	0%	0%	0%	OS logging	SIEM	All users
3	Actions on Objectives	AO	Account breached	AO-ACC	AO-ACC-02	Detect Account Manipulation	https://attack.mitre.org/wiki/Technique/T1098	0%	0%	0%	0%	0%	OS logging	SIEM	All users
3	Actions on Objectives	AO	Account breached	AO-ACC	AO-ACC-03	Detect Unauthorized Account Creation	https://attack.mitre.org/wiki/Technique/T1136	0%	0%	0%	0%	0%	OS logging	SIEM	Administrators
1	Actions on Objectives	AO	Internal Brute force	AO-BRU	AO-BRU-01	Detect Brute Force Attack	https://attack.mitre.org/wiki/Technique/T1110	0%	0%	0%	0%	0%	Database log	SIEM	All users
8	Actions on Objectives	AO	Data collection	AO-COL	AO-COL-01	Detect Automated Collection	https://attack.mitre.org/wiki/Technique/T1119	0%	0%	0%	0%	0%	Application log	SIEM	All users
8	Actions on Objectives	AO	Data collection	AO-COL	AO-COL-02	Detect Anomalous Clipboard Data	https://attack.mitre.org/wiki/Technique/T1115	0%	0%	0%	0%	0%	Application log	SIEM	All users
8	Actions on Objectives	AO	Data collection	AO-COL	AO-COL-03	Detect Staged Data	https://attack.mitre.org/wiki/Technique/T1074	0%	0%	0%	0%	0%	Application log	SIEM	All users

Figure 39: Example of the generated L3 use case for the ROS-based scenario

This process was followed to extract all the L3 UCs for the ROS-based scenario.

IMPORTANT NOTE: The exact characteristics of the system under analysis are considered during this step. This means that any configuration, communication protocol, firewall rules, etc. have to be considered here when building the L3 UCs.

IMPORTANT NOTE: It is worth noticing that the value of the effectiveness, implementation and coverage metrics has to be set manually for each L3 UC. This means that it depends hugely on the system under analysis. It is recommended that the row “Comments” on the L3 UC page is used to specify the reason behind each value of each metric.

IMPORTANT NOTE: In this step, the only cells that can be modified are those that belong the columns from “F” to “K”, and from “N” to “Q”. The remaining cells in the L3 UC page are automatically filled. If more L3 Use Cases are needed, they should be added after row 170. The rest of the document should be manually updated to reflect this change.

7.9. Interpretation of the Metrics

The MaGMa Use Case Framework provides two metrics that help to improve our use cases:

- **Weight:** This metric is a calculated overall score of the effectiveness, implementation, and coverage for each use case. A low value of the weight indicates that one of the three metrics (effectiveness, implementation, or coverage) has a low value that should be improved. High values of this metric are preferred, indicating that all three metrics have a high value.
- **Potential:** This metric is a calculated value that indicates how much improvement can be gained by investing in coverage and implementation. Thus, use cases that have a high effectiveness, but low coverage and implementation are assigned with a high potential value. Low values of this metric are preferred, because they indicate that there is no possible enhancement that can be done to the implementation and the coverage.

To interpret these metrics, MaGMa uses a diverging color pallet from red (low values) to green (high values). Next figure shows the colors used by MaGMa.

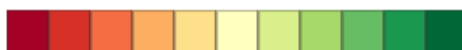


Figure 40: Colour scale used in MaGMa

The color scale used in MaGMa to represent the result of each metric is presented in Figure 40. Red color represent low values of the metric, while green represent high values.

Although this scale seems useful, it fails to help interpreting their value because red color values (low values) do not always imply a negative interpretation. This can be easily seen when every metric has its optimal value. Figure 41 shows an example of this, where the effectiveness, implementation, coverage, and weight are highlighted in green (indicating a high value, and implicitly, a good value). Meanwhile, the potential is highlighted in red (indicating a low value, and implicitly, a non-desirable value). When in reality, a value of 0 % for the potential is the optimal. Even though the Potential metric is in red, its value corresponds with the most appropriated value this metric can have.

For this reason, it is critical to understand that the color scale for the potential is inverted and low values are desirable.

Effectiveness %	Implementation %	Coverage %	Weight (eff*impl*cvrge)	Potential (eff-weight)
100%	100%	100%	100%	0%

Figure 41: Example of an optimal deployed use case

In summary, high values for every metric are targeted, except for the potential, where low values are preferred. An example of generated data can be presented in the following figure.

	Effectiveness %	Implementation %	Coverage %	Weight (eff*impl*cvrge)	Potential (eff-weight)
1					
22	74%	68%	86%	43%	31%
23	90%	52%	92%	43%	47%
24	7%	93%	69%	4%	3%
25	42%	2%	24%	0%	42%
26	86%	39%	80%	27%	59%

Figure 42: Generated dataset of the metrics defined in MaGMa

The values shown in row 22 in Figure 42 are relatively balanced. Nevertheless, it is observed that the potential is 31 %, indicating that the implementation or the coverage can be enhanced. More precisely, the potential is indicating that the implementation should be improved as the implementation is 68 % (lower than the coverage). The weight has a low value, indicating that the effectiveness, the implementation, or the coverage should be improved. In this case, despite the fact that the implementation should be enhanced, it is supposed that the weight indicates the improve of the effectiveness of this detection mechanism.

It is worth noticing that row 23 in Figure 42 has the same weight as row 22 (43 %). However, based on the values, it can be concluded that the problematic value here is the implementation of the detection mechanism.

Finally, analyzing the values in row 24 of the above figure, the low value of the weight highlights the need to improve mainly the effectiveness, because the potential has a low value, indicating that neither the implementation, nor the coverage need improvement.

7.10. References

1. K. Stouffer, V. Pillitteri , S. Lightman , M. Abrams and A. Hahn , Guide to Industrial Control Systems (ICS) Security, U.S. Department of Commerce, 2015.
2. International Electrotechnical Commission (IEC), Industrial communication networks - network and system security, Geneva, 2018.
3. V. Mayoral Vilches, I. Apellaniz Aparicio, U. Ayucar Carbajo and E. Gil Uriarte , White Paper - Red Team: Robot Cybersecurity for Industrial ROS Applications, Alias Robotics.
4. R. van Os, F. Ladan, T. van Casteren, R. Toornstra, R. Metsemakers, L. Nieuwenhuize and H. Grotenhuis, MaGMA Use Case Framework, FI-ISAC NL, 2017.
5. Canonical, “Securing ROS robotics platforms,” March 2020. [Online]. Available: <https://ubuntu.com/engage/securing-ros-on-robotics-platforms-whitepaper>. [Accessed 10 June 2022].
6. Universal Robots, <https://www.universal-robots.com/>
7. MaGMA UCF tool excel file, [Magma UCF tool](#).