Secure management of IoT devices lifecycle through identities, trust and distributed ledgers

D4.5 Intrusion detection for IoT-based context and networks

Document Summary Information

<table>
<thead>
<tr>
<th>Grant Agreement No</th>
<th>101020416</th>
<th>Acronym</th>
<th>ERATOSTHENES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Title</td>
<td>Secure management of IoT devices lifecycle through identities, trust and distributed ledgers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start Date</td>
<td>01/10/2021</td>
<td>Duration</td>
<td>42 months</td>
</tr>
<tr>
<td>Project URL</td>
<td><a href="http://www.eratosthenes-project.eu">www.eratosthenes-project.eu</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deliverable</td>
<td>D4.5 Intrusion detection for IoT-based context and networks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work Package</td>
<td>WP5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contractual due date</td>
<td>June 30, 2023</td>
<td>Actual submission date</td>
<td>27/06/2023</td>
</tr>
<tr>
<td>Nature</td>
<td>Other</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dissemination Level</td>
<td>Public</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Responsible author</td>
<td>Rosella Omana Mancilla</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lead Beneficiary</td>
<td>ENG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authors</td>
<td>Rosella Omana Mancilla, Francesca Costantino (ENG), Juan Manuel Vera Diaz (ATOS)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internal reviewers</td>
<td>Jesús García Rodríguez (UMU), Konstantinos Krilakis (EUL), Sokrates Vavilis (INLE)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Grant Agreement No 101020416.
Revision history (including peer reviewing & quality control)

<table>
<thead>
<tr>
<th>Version</th>
<th>Issue Date</th>
<th>% Complete</th>
<th>Changes</th>
<th>Contributor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0.0</td>
<td>26/04/23</td>
<td>2%</td>
<td>TOC</td>
<td>ENG</td>
</tr>
<tr>
<td>v0.1</td>
<td>17/05/23</td>
<td>80%</td>
<td>Initial content</td>
<td>ENG</td>
</tr>
<tr>
<td>v0.2</td>
<td>19/05/23</td>
<td>90%</td>
<td>Input and comments from ATOS, additional</td>
<td>ENG, ATOS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>content</td>
<td></td>
</tr>
<tr>
<td>v0.3</td>
<td>01/06/23</td>
<td>100%</td>
<td>Input and comments from UMU, additional</td>
<td>ENG, UMU</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>content</td>
<td></td>
</tr>
<tr>
<td>v0.4</td>
<td>05/06/23</td>
<td>100%</td>
<td>Inputs from UPRC</td>
<td>UPRC</td>
</tr>
<tr>
<td>v0.5</td>
<td>26/06/23</td>
<td>100%</td>
<td>Addressed review comments</td>
<td>ENG</td>
</tr>
<tr>
<td>v1.0</td>
<td>27/06/23</td>
<td>100%</td>
<td>Addressed quality review and other fixes.</td>
<td>ENG</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Final version reached.</td>
<td></td>
</tr>
</tbody>
</table>

Disclaimer

The content of the publication herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the documents is believed to be accurate, the authors(s) or any other participant in the ERATOSTHENES consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the ERATOSTHENES Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing neither the ERATOSTHENES Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.

Copyright message

© ERATOSTHENES Consortium, 2020-2025. This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both. Reproduction is authorised provided the source is acknowledged.
Table of Contents

1 Executive Summary ........................................................................................................................................... 6
2 Introduction ..................................................................................................................................................... 7
  2.1 Mapping ERATOSTHENES Outputs ............................................................................................................. 7
  2.2 Deliverable Overview and Report Structure ................................................................................................. 9
3 Architecture Orientation and Industrial Requirements ...................................................................................... 10
  3.1 Architectural Positioning and design decisions ............................................................................................ 10
  3.2 Business, Industrial Positioning and End-User Requirements ..................................................................... 11
  3.3 Methodology ............................................................................................................................................... 13
  3.4 Code Availability ....................................................................................................................................... 14
4 Network Intrusion Prevention Detection System for IoT-based Environments ..................................................... 15
  4.1 Architecture .............................................................................................................................................. 15
  4.2 Implementation ......................................................................................................................................... 16
    4.2.1 Engine .................................................................................................................................................. 17
    4.2.2 Anomaly Detection Inspector ............................................................................................................. 19
    4.2.3 Alert GUI ............................................................................................................................................. 21
    4.2.4 Threat/Rule Manager .......................................................................................................................... 22
    4.2.5 Score Calculator .................................................................................................................................... 24
  4.3 Interconnection with ERATOSTHENES tools ............................................................................................. 24
    4.3.1 Sequence diagrams ............................................................................................................................ 25
  4.4 Deployment ............................................................................................................................................... 27
5 Research and Scientific Innovation ..................................................................................................................... 29
  5.1 Anomaly Detection ................................................................................................................................... 29
  5.2 SOTA - Machine learning paradigms ........................................................................................................... 29
  5.3 Beyond SOTA .............................................................................................................................................. 30
    5.3.1 Testing Results ..................................................................................................................................... 31
6 Conclusions ..................................................................................................................................................... 41
7 References ....................................................................................................................................................... 42

ANNEX I: Open-source Security Vulnerabilities .................................................................................................. 44

List of Figures

Figure 1 – Current ERATOSTHENES architecture, after POC evaluation ............................................................... 10
Figure 2 – IDS subcomponents ............................................................................................................................ 16
Figure 3 – Alerts generated by the Engine .......................................................................................................... 19
Figure 4 – ADI submodules .................................................................................................................................. 20
Figure 5 - Alert GUI Dashboard .......................................................................................................................... 22
Figure 6 - Sequence diagram for IDS functionalities ........................................................................................ 25
Figure 7 - Other mitigation sequence diagram .................................................................................................. 26
Figure 8 - ADI model forwarding to FedLPY ..................................................................................................... 27
Figure 9 - docker-compose.yml for the IDS/IPS solution .................................................................................... 28
List of Tables

Table 1: Adherence to ERATOSTHENES GA Deliverable & Tasks Descriptions.................................................................7
Table 2 – Requirements from D1.2 ........................................................................................................................................12
Table 3 – Threat classifications...........................................................................................................................................17
Table 4 - MQTT Topics where Threat/Rule Manager subscribes or write...........................................................................22
Table 5 – MQTT Topics where the Score Calculator subscribes or writes...........................................................................24
Table 6 - List of dependencies...............................................................................................................................................27
Table 7 - Attack types of NF-ToN-IoT dataset .......................................................................................................................32
Table 8 - NF-ToN-IoT features ...............................................................................................................................................33
Table 9 - Attack types in IoT-23 combined dataset .............................................................................................................33
Table 10 - IoT-23 list of features........................................................................................................................................34
Table 11 - Summary of tests..................................................................................................................................................40
Table 12 - Known vulnerabilities .........................................................................................................................................44
### Glossary of terms and abbreviations used

<table>
<thead>
<tr>
<th>Abbreviation / Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>Anomaly Detection</td>
</tr>
<tr>
<td>ADFA</td>
<td>Australian Defence Force Academy</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial Intelligence</td>
</tr>
<tr>
<td>ADI</td>
<td>Anomaly Detection Inspector</td>
</tr>
<tr>
<td>CTI</td>
<td>Cyber-Threat Information</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>Density-Based Spatial Clustering of Applications with Noise</td>
</tr>
<tr>
<td>DLT</td>
<td>Distributed Ledger Technology</td>
</tr>
<tr>
<td>DOW</td>
<td>Document Of Work</td>
</tr>
<tr>
<td>ELK</td>
<td>Elasticsearch, Logstash and Kibana</td>
</tr>
<tr>
<td>IDS/IPS</td>
<td>Intrusion Detection System/Intrusion Prevention System</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet Of Things</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ML</td>
<td>Machine Learning</td>
</tr>
<tr>
<td>MVP</td>
<td>Minimum Viable Product</td>
</tr>
<tr>
<td>MUD</td>
<td>Manufacturer Usage Description</td>
</tr>
<tr>
<td>NF-ToN-IoT</td>
<td><strong>NetFlow</strong> Telemetry datasets of <strong>IoT</strong> and IIoT sensors, <strong>Operating</strong> systems datasets of Windows 7 and 10 as well as Ubuntu 14 and 18 <strong>TLS</strong> and <strong>Network</strong> traffic datasets</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>NIS</td>
<td>Network and Information Security</td>
</tr>
<tr>
<td>POC</td>
<td>Proof of Concept</td>
</tr>
<tr>
<td>SEIT</td>
<td>School of Engineering and Information Technology</td>
</tr>
<tr>
<td>TMB</td>
<td>Trust Manager &amp; Broker</td>
</tr>
<tr>
<td>WP</td>
<td>Work Package</td>
</tr>
</tbody>
</table>
1 Executive Summary

This deliverable “D4.5 Intrusion detection for IoT-based context and networks” concerns the description of the initial activities carried out in task T4.5 for the design, implementation, and deployment of the IoT-context-based detection capability for the ERATOSTHENES project, as well as the innovation that the solution can bring to the state of art in Intrusion Detection.

The work reported in this document required close interaction with other Work Packages:

- WP1 for the requirements, the use-cases and the architecture
- WP2 for the cooperation and integration with the TMB modules
- Other tasks in WP4 for the cooperation and integration with the FedLpy and the device lifecycle security
- WP5 for the pilots’ input and feedback

The proposed solution will be provided in three main versions and this document reports the details of the first one, next will be reported in deliverables D4.7 AI Threat Analysis Models and Intrusion Detection for IoT Networks M29 and D4.8 Final Version of DLTbased Trust Framework and AI threat analysis models M36.

The first version hereafter reported, includes partial implementation of the following functionalities to enhance IDS capabilities:

- CTI Sharing
- Anomaly Detection
2 Introduction

This document aims at reporting and tracking the information needed to effectively define the architecture, system design and implementation of the Intrusion Detection System/Intrusion Prevention System (IDS/IPS) developed in the context of the ERATOSTHENES project, within task T4.5.

The document is provided in month M21 of the project, and it will be refined with WP1 output, and other tasks, following an incremental and iterative methodology, as well as the feedback from the first round of Pilots’ demonstration (WP5).

The design and development are coordinated with the help of other tasks’ output. Specifically:

- task T1.2 provides the requirements and the use-cases
- task T1.4 provides the architecture
- task T2.1 provides input for cooperation and integration with the TMB modules
- task T4.4 provides input for the cooperation and integration with the FedLpy
- task T4.6 provides input for the cooperation and integration with the device lifecycle security
- task T5.1 provides input for the integration
- task T5.2, T5.3, T5.4, T5.6 provides inputs from the pilots

2.1 Mapping ERATOSTHENES Outputs

Table 1: Adherence to ERATOSTHENES GA Deliverable & Tasks Descriptions

<table>
<thead>
<tr>
<th>ERATOSTHENES GA Component Title</th>
<th>ERATOSTHENES GA Component Outline</th>
<th>Respective Document Chapter(s)</th>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELIVERABLE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D4.5 Intrusion Detection for IoT-based context and networks</td>
<td>This deliverable includes the first version of the development of IDS as a result from Task 4.5</td>
<td>Chapter 4, Chapter 5</td>
<td>Chapter 4 describes in detail the initial design system specifications and implementation. Chapter 5 describes the research and studies made to enhance detection capabilities.</td>
</tr>
<tr>
<td>TASKS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T4.5 Intrusion detection for IoT-based context and networks</td>
<td>Research made on IoT traffic have reported that if zooming in on an IoT device traffic load during a short interval, activity patterns, signalling, protocols, etc. emerge. (E.g. active/sleep communication patterns). From this assumption, the usage of M-L techniques, like clustering</td>
<td>Chapter 4, Chapter 5</td>
<td>Chapter 4 describes in detail the initial design system specifications and implementation. Chapter 5 describes the research and studies made to enhance detection capabilities.</td>
</tr>
</tbody>
</table>
algorithms (e.g. K-NN), could increase the capabilities of incidents/breach detection in a constrained smart environment. This task will focus on the implementation of the IoT based context detection capability. The implementation process will start from the studies made in WP1 on the SOTA of IDS and IoT IDS. The detection capability for the IoT context will be implemented with the application of several machine learning, and not, techniques to enhance traditional IDS capabilities: the best solution will be chosen among different combinations, in terms of detection rate and overall accuracy. The task will start with implementing the algorithms into a big-data analytics workflow platform and testing them using available opensource dataset such as KDD Cup 99, or better, IoT network traffic such as IoT-23 (and others from T4.4). The classification resulted by the chosen algorithm will then help in enhancing the traditional IDS with IoT-based detection capabilities. The new IoT-context based IDS will monitor, analyse and refine in semi-real time the classifications with real traffic data, detect malicious activities and raise alerts through the ERATOSTHENES framework. The solution will monitor the network, and the detection capability will be tested in relevant environments defined in WP5. Moreover, there will
be close interaction between T4.4 and T4.5 in order to enhance more both the detection capability and threat analysis: detected attacks, especially related to zero-day vulnerabilities or new types of attack, in a constrained environment, will surely help the threat analysis made in task T4.4, on the other hand, the federated threat analysis models can enhance the algorithm to detect anomalies in a restricted network. The results will be reported in D4.5, in D4.7 (updated version) and in the final version D4.8.

2.2 Deliverable Overview and Report Structure

In the list below the document structure is reported:

- Chapter 2: Introduction
- Chapter 3: Solution Positioning in the ERATOSTHENES Architecture, Business Value, the Methodology and some requirements
- Chapter 4: Network Intrusion Detection, with the preliminary design and implementation of the solution, a high-level architecture is depicted
- Chapter 5: The innovation and the studies made to design the solution
- Chapter 6: Conclusion
- Chapter 7: References

This document reports the first version of the proposed solution, next versions will be the updated version which will be reported in the deliverable D4.7 AI Threat Analysis Models and Intrusion Detection for IoT Networks M29, and the final e complete version, reported in D4.8 Final Version of DLTbased Trust Framework and AI threat analysis models M36.
3 Architecture Orientation and Industrial Requirements

3.1 Architectural Positioning and design decisions

The ERATOSTHENES project’s purpose is to enhance the cybersecurity of IoT ecosystems, in particular, it focuses on the cybersecurity lifecycle management for IoT devices.

The framework proposed includes different sub-modules, some based on available solutions, that have to be developed/adapted/enhanced, during the project life, then demonstrated in the context of the project’s pilots.

The current architecture grabbed from WP1 activities, is depicted in Figure 1, it is refined with advances on the ERATOSTHENES modules (initial version is available in [1]).

The IDS/IPS is part of the Trust Manager and Broker (TMB) sub-module called “Monitoring, IDS”. The aim of the sub-module is to enhance the security of the infrastructure, IoT based, by identifying threats, possible threats and/or potential zero-day attacks.

The solution reported was not part of the Proof of Concept (POC) implementation/instantiation, and it will be added and integrated with what was already included in the ERATOSTHENES framework (POC version), see [2]: initial integration design is described in section 4.3.
3.2 Business, Industrial Positioning and End-User Requirements

To enhance the strength of the IoT ecosystems, and follow the NIST guidelines [3] and NIS directive [4], it is extremely important to establish monitoring and detection capability to spot the rise of a cybersecurity event.

The NIST defines intrusion detection as “the process of monitoring the events occurring in a computer system or network and analysing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices.” [5].

Monitoring and analysing the traffic in the network will ensure that each identified and authenticated device is generating legitimate traffic, and is not a shadow IoT device. Incidents are not always malicious, and an employee could try to access a restricted domain without permission, by mistake. Such events must be reported to make sure that the ecosystem can react and remains protected.

Intrusion Detection Systems (IDS) play an important role in helping establish robust and comprehensive security. An IDS is a component that monitors network traffic for suspicious activity and alerts when such activity is discovered. Some IDSs can take action when a malicious activity or anomalous traffic is detected, including blocking traffic sent from suspicious Internet Protocol (IP) addresses, in this case, we call it Intrusion Prevention System (IPS).

Today’s ecosystems created considerable challenges to traditional Intrusion Detection Systems: massive network data traffic, detection efficiency and real-time detection are just a few of them with high impact on the definition of modern solutions.

And, as suggested in directive 51 [4], which encourages Member States the adoption of innovative technologies, such as Artificial Intelligence, the solution proposed for ERATOSTHENES is an Intrusion Detection System, that can be configured as IPS too, enhanced with the analysis of IoT traffic and detection of attacks, to be integrated within the project framework: it uses Machine Learning techniques to adequately and efficiently classify normal behaviours or habits of devices connected from new types of attack.

Moreover, the adoption of an open-source solution for the signature-based engine contributes to a higher level of openness and transparency and can have a positive impact on the efficiency of proposed innovation, as stated in Directive 52 [4].

The adoption of Cyber-Threat Information Sharing Agents (CTI Sharing Agents) is also foreseen to notify external stakeholders whenever an incident occurs and to contribute to ensuring that the detection capabilities used are finely tuned and the solution can respond to new vulnerabilities and changes in the environment.

In fact, from D1.2 Use Cases, Requirements and Methodological Framework, the following have been defined as research priorities related to ERATOSTHENES objectives, and further studied to be included in the proposed solution:

- Cyberthreat information sharing to CERTS/CSRTs interface to inform for cyberattacks and threats (IDS events)
- Monitoring the device behaviour for suspicious and anomalous indicators in the V2V/V2I communications through trust agents and network-based IDS to identify possible attackers or malfunctioning devices. For example, possible actions for suspicious activities are limiting the capabilities of the device, denying its entry or removing it from the network.

And corresponding requirements are reported in Table 2.
<table>
<thead>
<tr>
<th>Req ID</th>
<th>Description</th>
<th>Importance</th>
<th>Rationale</th>
<th>Validation Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1_FR_01</td>
<td>Trusted information exchange with CERT/CSIRTs for automotive sector (or similar) vulnerabilities reporting</td>
<td>M</td>
<td>Participate in information-sharing platforms to report vulnerabilities and receive timely and critical information about current cyber threats and vulnerabilities from public and private partners of the automotive (or other sectors)</td>
<td>Information pushing towards CERT/CSIRTs on recent vulnerabilities in the automotive field</td>
</tr>
<tr>
<td>P3_FR_01</td>
<td></td>
<td></td>
<td></td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>P1_FR_05</td>
<td>The infrastructure is monitored, and network traffic is analysed to detect intrusions</td>
<td>M</td>
<td>The integrity of the communications between the vehicles and the infrastructure is ensured by some intrusion/anomaly detection</td>
<td>Attacks are detected by ML techniques and anomalies are notified and available</td>
</tr>
<tr>
<td>P2_FR_20</td>
<td></td>
<td></td>
<td></td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>P3_FR_03</td>
<td></td>
<td></td>
<td></td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>P1_NFR_02</td>
<td>Improve detection of malicious messaging from compromised/unsecured actors</td>
<td>M</td>
<td>Improved detection of not identified, suspicious or considered not secure devices in the network</td>
<td>Simulated malicious messaging to automotive devices</td>
</tr>
<tr>
<td>P1_NFR_12</td>
<td>Accuracy (or precision) on detecting attacks and/or anomalies</td>
<td>S</td>
<td>Accuracy of attacks detection on IoT devices must be very high</td>
<td>Simulated attacks</td>
</tr>
<tr>
<td>P2_NFR_11</td>
<td></td>
<td></td>
<td></td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>P3_NFR_10</td>
<td></td>
<td></td>
<td></td>
<td>------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EXT_FR_01</td>
<td>IDS must be able to generate reports</td>
<td>M</td>
<td>CTI reports will be key for enhanced security across the whole</td>
<td>Establish a monitoring scenario where CTI reports will be generated</td>
</tr>
</tbody>
</table>
### 3.3 Methodology

The main strategy used in Task 4.5, and within the whole project, is based on **Agile** principles and **Minimum Viable Product** strategy (MVP), therefore the process to achieve the solution follows an incremental and iterative methodology. As an additional input on procedures and methodology, the task follows standard ISO 27039.

First, the preliminary architecture defined in the Document of Work is analysed to elicit an initial and high-level list of **requirements**. Then, the use cases defined in WP2 are analysed to retrieve more requirements. Lastly, during WP5 meetings and plenary meetings, initial requirements that are more specific to pilots’ needs are elicited.

From the initial phase, the architecture of the solution as well as the functionalities are defined: functionalities are also described through sequence diagrams since they foresee the interaction with other ERATOSTHENES solutions.

Then the functionalities are built based on the output provided by the **design** phase and tested with Unit Tests and Integration tests.

At this point, the solution will be **presented/demonstrated** to the pilots to have initial feedback on the implementation. The feedback will be collected and translated into requirements updates and/or new requirements. At this stage, pilots’ specifications and pilots’ data (pcap datasets) are also requested to test the solution, in particular the Anomaly Detection Inspector, with pilots’ network traffic.

This process will be carried out during the whole project, iteratively and it is in line with the scheduling of the Pilots activities: see [2] – Figure 1 – ERATOSTHENES workflow from the DOW.

Before the Pilot demonstration phase, the solution is **integrated into the ERATOSTHENES framework** and integration and system tests are performed.

It is worth mentioning that by the time this document has been released, pilots have just started defining details on their infrastructure, technical specifications as well as user-stories, therefore this document reports analysis made on available information on pilots.

Also, changes may occur on the following report due to updates on other work packages, in particular in WP2.
3.4 Code Availability

The code for the solution described in this document is available in the project’s repository, dev branch.

https://ci-cysec.eng.it/gitlab/ERATOSTHENES/wp4/ids-ips
4 Network Intrusion Prevention Detection System for IoT-based Environments

The solution proposed for ERATOSTHENES is based on a previous project result\(^1\), where detection capabilities have been explored and studied to design a new approach in incidents identification: ENG focuses the research on the adoption of anomaly detection capabilities to enhance current SOTA on Intrusion Detection.

ERATOSTHENES project is the perfect context where the studies can be followed-up.

The Intrusion Detection and Prevention System is a signature-based solution which can identify threats based on signatures on previously detected attacks, and the adaptation envisioned in this project, based on WP1 output, requirements listed in Table 2 and pilot collaboration, is focused on the following high-level activities in the IoT-based ecosystem:

- CTI sharing – sharing detected threats, updates detection rules
- Anomaly detection – identification of potential misbehaviour or new attacks based on prerecorded traffic
- Device Trust Monitoring – change devices’ Trust level based on generated traffic

To implement the just mentioned activities, in the next subchapters, the initial architecture, implementation and possible deployments are reported.

4.1 Architecture

The IDS solution designed and developed in the context of ERATOSTHENES is composed of the following high-level sub-components, depicted in Figure 2:

- Engine
- Threat/Rule Manager
- Anomaly Detection Inspector (ADI)
- Score Calculator
- Alert GUI

Engine

The *Engine* is an open-source IDS that is widely known and used, also in research activities and papers. *SNORT*\(^2\) has been selected and enhanced because of different features that make it useful for network admins to monitor the ecosystem and detect malicious activity. Some of the reasons why SNORT is the selected one are listed hereafter:

- Real-time traffic monitoring
- Can be installed in any network environment
- Open Source [6]
- Rules are easy to implement
- Up-to-date free community rules
- Prevention can be enabled
- Great community support
- Plug-in framework, make key components pluggable (and 200+ plugins)

---

\(^1\) CITYSCAPE project, https://www.cityscape-project.eu/
\(^2\) https://www.snort.org/
• Multi-threading for packet processing (SNORT version 3)

**Threat/Rule Manager**

This sub-module is in charge of sharing detected threats and updating the rule set for the *Engine*, based on other domain threats or another request from some decision support system and/or mitigation executor. In other words, is the component that implements the *CTI Sharing capability*.

**Anomaly Detection Inspector**

This sub-module implements the *Anomaly detection* capability that ERATOSTHENES is proposing. It uses Machine Learning techniques to potentially detect possible threats or new types of attack. Details on the research and innovation are provided in the next chapter 5.

**Score Calculator**

This component is implementing the *Device Trust Monitoring* capability. Based on the “bad” traffic generated by a specific device, a value is provided to the Trust Manager(TM), to request the update of the device Trust Score (details are available in D2.1 Trust Broker Mechanism).

**Alert GUI**

The selected *Engine* has no incorporated graphical user interface (GUI), therefore an interface to display alerts has been designed as part of the solution.

---

**4.2 Implementation**

This section reports more details on the implementation of the IDS sub-components since month M20.
4.2.1 Engine

The usage of SNORT lets the solution detects and take actions following a signature-based modality: it identifies each packet with a class using community rules and locally defined rules (specific to the ecosystem).

The solution inherits the macro classifications by SNORT, available in Table 3, each of them is identified with a set of rules. Each class is defined with a short name, a description and the priority level

- 1 – High,
- 2 – Medium,
- 3 – Low,
- 4 – Very low

<table>
<thead>
<tr>
<th>short name</th>
<th>short description</th>
<th>priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>attempted-user</td>
<td>Attempted User Privilege Gain</td>
<td>1</td>
</tr>
<tr>
<td>unsuccessful-user</td>
<td>Unsuccessful User Privilege Gain</td>
<td>1</td>
</tr>
<tr>
<td>successful-user</td>
<td>Successful User Privilege Gain</td>
<td>1</td>
</tr>
<tr>
<td>attempted-admin</td>
<td>Attempted Administrator Privilege Gain</td>
<td>1</td>
</tr>
<tr>
<td>successful-admin</td>
<td>Successful Administrator Privilege Gain</td>
<td>1</td>
</tr>
<tr>
<td>shellcode-detect</td>
<td>Executable Code was Detected</td>
<td>1</td>
</tr>
<tr>
<td>trojan-activity</td>
<td>A Network Trojan was Detected</td>
<td>1</td>
</tr>
<tr>
<td>web-application-attack</td>
<td>Web Application Attack</td>
<td>1</td>
</tr>
<tr>
<td>inappropriate-content</td>
<td>Inappropriate Content was Detected</td>
<td>1</td>
</tr>
<tr>
<td>policy-violation</td>
<td>Potential Corporate Privacy Violation</td>
<td>1</td>
</tr>
<tr>
<td>file-format</td>
<td>Known malicious file or file-based exploit</td>
<td>1</td>
</tr>
<tr>
<td>malware-cnc</td>
<td>Known malware command and control traffic</td>
<td>1</td>
</tr>
<tr>
<td>client-side-exploit</td>
<td>Known client-side exploit attempt</td>
<td>1</td>
</tr>
<tr>
<td>bad-unknown</td>
<td>Potentially Bad Traffic</td>
<td>2</td>
</tr>
<tr>
<td>attempted-recon</td>
<td>Attempted Information Leak</td>
<td>2</td>
</tr>
<tr>
<td>successful-recon-limited</td>
<td>Information Leak</td>
<td>2</td>
</tr>
<tr>
<td>short name</td>
<td>short description</td>
<td>priority</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>--------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>successful-recon-largscale</td>
<td>Large Scale Information Leak</td>
<td>2</td>
</tr>
<tr>
<td>attempted-dos</td>
<td>Attempted Denial of Service</td>
<td>2</td>
</tr>
<tr>
<td>successful-dos</td>
<td>Denial of Service</td>
<td>2</td>
</tr>
<tr>
<td>rpc-portmap-decode</td>
<td>Decode of an RPC Query</td>
<td>2</td>
</tr>
<tr>
<td>suspicious-filename-detect</td>
<td>A Suspicious Filename was Detected</td>
<td>2</td>
</tr>
<tr>
<td>suspicious-login</td>
<td>An Attempted Login Using a Suspicious Username was Detected</td>
<td>2</td>
</tr>
<tr>
<td>system-call-detect</td>
<td>A System Call was Detected</td>
<td>2</td>
</tr>
<tr>
<td>unusual-client-port-connection</td>
<td>A Client was Using an Unusual Port</td>
<td>2</td>
</tr>
<tr>
<td>denial-of-service</td>
<td>Detection of a Denial of Service Attack</td>
<td>2</td>
</tr>
<tr>
<td>non-standard-protocol</td>
<td>Detection of a Non-Standard Protocol or Event</td>
<td>2</td>
</tr>
<tr>
<td>web-application-activity</td>
<td>Access to a Potentially Vulnerable Web Application</td>
<td>2</td>
</tr>
<tr>
<td>misc-attack</td>
<td>Misc Attack</td>
<td>2</td>
</tr>
<tr>
<td>default-login-attempt</td>
<td>Attempt to Login By a Default Username and Password</td>
<td>2</td>
</tr>
<tr>
<td>sdf</td>
<td>Sensitive Data was Transmitted Across the Network</td>
<td>2</td>
</tr>
<tr>
<td>not-suspicious</td>
<td>Not Suspicious Traffic</td>
<td>3</td>
</tr>
<tr>
<td>unknown</td>
<td>Unknown Traffic</td>
<td>3</td>
</tr>
<tr>
<td>string-detect</td>
<td>A Suspicious String was Detected</td>
<td>3</td>
</tr>
<tr>
<td>network-scan</td>
<td>Detection of a Network Scan</td>
<td>3</td>
</tr>
<tr>
<td>protocol-command-decode</td>
<td>Generic Protocol Command Decode</td>
<td>3</td>
</tr>
<tr>
<td>misc-activity</td>
<td>Misc activity</td>
<td>3</td>
</tr>
<tr>
<td>icmp-event</td>
<td>Generic ICMP event</td>
<td>3</td>
</tr>
<tr>
<td>tcp-connection</td>
<td>A TCP Connection was Detected</td>
<td>4</td>
</tr>
</tbody>
</table>
Generated alerts are collected into a local file (format can be configured in the lua file): an example of alerts is depicted in Figure 3.

![Generated alerts example](https://example.com/generated_alerts.png)

**Figure 3 – Alerts generated by the Engine**

The Engine can be configured by updating the configuration file which is available in `/usr/local/etc/snort/snort.lua`:

- Output
- Detection rules
- Inspectors
- Other configuration file
- Etc...

### 4.2.2 Anomaly Detection Inspector

This sub-module aims to generate the Anomaly Detection Model which will be used by the *Engine* to enhance its detection capability: the model is trained with traffic that includes normal behaviour and potentially detect zero-day attacks and misbehaviour.

The Anomaly Detection Inspector is implemented in Python [7] version 3.9, selected mainly for its capabilities in implementing Machine Learning and Deep Learning algorithms, using libraries such as Scikit-learn [8].

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python’s elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

https://docs.python.org/3.9/tutorial/index.html

The ADI is divided into four parts:

- *model* package contains the ADI implementation
- *Security* package contains the implementation of authentication and authorization to use the API
- **API** contains the services to be exposed to request generation of an anomaly detection model and to test the ADI implementation
- **Test** contains unit-testing

The **model** package contains the following classes:

- *data_loader.py*  # for the loading of data
- *data_transformer.py*  # for the various transformation of data
- *clu_clf_model.py*  # contains the clustering and classification

The package **API** is implemented with FastAPI\(^3\) [9] which is a modern, fast (high-performance), web framework for building APIs with Python version 3.6 or higher.

The services exposed, for the current are:

- POST /create_user  # for the registration of a new user
- POST /token  # to log in as a user
- PUT /updateanomalymodel  # to train the ADP and create a detection model
- PUT /predict  # to classify records
- PUT /compare  # to compare results of different classification algorithms given a training dataset and a test labelled dataset

---

\(^3\) Licensed under the terms of MIT license  [https://fastapi.tiangolo.com/#license](https://fastapi.tiangolo.com/#license)
With the updateanomaly service a detection model is created and saved locally (in a joblib format [10]), this one must be used in the future by the Engine to enhance its capabilities in detecting anomalies. The integration with the Engine is foreseen as part of the next versions, details will be available in [11] or [12].

The algorithm used to create the model is based on ML techniques. The reasons and studies behind the selection of ML techniques are provided in Chapter 5. Hereafter is a high-level description of the algorithm.

1. train_set = load dataset
2. for r in train_set
3. TransformI
4. normalize(train_set)
5. clustered_set = DBSCAN.fit_predict(train_set)
6. binary_set = clustered_set.combine(IoT-based-known-dataset)
7. for n in binary_set
8. if(n.cluster == -1)
9. n.label = 1
10. else
11. n.label = 0
12. ad_model = CLASS_ALG.fit(binary_set)
13. save(ad_model)

### 4.2.3 Alert GUI

This sub-module is added to have a simple Graphical User Interface to display alerts into a user-friendly Dashboard.

It is implemented as an Elastic stack, aka ELK (Elasticsearch, Logstash and Kibana) [13] which are three open-source projects:

- **Elasticsearch is a search and analytics engine**
- **Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch**
- **Kibana lets users visualize data with charts and graphs in Elasticsearch**. [13]

Logstash collects the alerts generated by the Engine, then they could be further elaborated with Elasticsearch, and lastly displayed in Kibana.

A Kibana Dashboard has been defined to view clear and useful information about the alerts, see Figure 5, and filters are also available to the user.

---

4 Basic License

© ERATOSTHENES, 2020
4.2.4 Threat/Rule Manager

The followings are the two main aims of this sub-module, see Figure 6:

1. To forward threats alerts to the CTI Agent
2. To receive new detection signature for the Engine

As for the ADI, this sub-module is implemented in Python. It is implemented as an MQTT client for Node.js, MQTT.js\(^5\), which subscribes and writes to a few channels in the Broker:

<table>
<thead>
<tr>
<th>MQTT topic</th>
<th>Description</th>
<th>Module role</th>
</tr>
</thead>
<tbody>
<tr>
<td>threatInfoSharing</td>
<td>Topic for sharing Threat data, (e.g. IDS -&gt; CTI Agent)</td>
<td>Writer and Subscriber</td>
</tr>
<tr>
<td>threatMudMSPL</td>
<td>Topic for sharing messages generated by the MUD Management Module translator, corresponding to an MSPL (medium-level security policy language) representation of the mitigations associated with a threat MUD file</td>
<td>Subscriber</td>
</tr>
<tr>
<td>mudMSPL</td>
<td>Topic for sharing message generated by the MUD Management Module translator, corresponding to an MSPL (medium-level security policy language)</td>
<td>Subscriber</td>
</tr>
</tbody>
</table>

\(^5\) https://github.com/mqttjs/MQTT.js
\(^6\) Topics names may change in next WP5 deliverables

© ERATOThENES, 2020
Currently, the alerts forwarded to the CTI Agent are those with high-level priority, an example of data exchanged is available hereafter. The data fields collect the information available, generated by the Engine, with the detection rule explicitly added. This one is typically a community SNORT3 rule or a locally generated rule.

```json
{
    "generator": "IDS",
    "type": "IDS event",
    "format": "erat:ids:event",
    "data": {
        "timest"mp": "11/25-16:09:11.650274",
        "pkt"_"um": 9367, "pr"to": "CP",
        "pkt"_"en": "stream_cp",
        "pkt"_"en": 16,
        "ir": "$2S", ## Direction C2S (client to server) S2C (server to client) UNK (unknown)
        "src_a"dr": "128.14.136"18", "src_p"rt": 4011",
        "src"_"ID": "did:erat:a...3",
        " dst_a"dr": "103.10.0."19", "dst_p"rt": 300",
        "dst"_"ID": "did:erat:a19",
        "serv"ce": "http",
        "prior"ty": ",
        "cl"ss": "Web Application Attack", ->check NIST, C"E
        "act"on": "allow",
        "rule"id": "1:6051":2",
        "r"ule": "...
    }
}
```

In the current version of the solution, not all the fields are provided and forwarded to the broker’s channel ThreatInfoSharing: the detection_rule explicitly defined is not yet added to the available data.

For the update of the signature set from the CTI Agent, the sub-component subscribes to the broker’s channel threatMudMSPL and mudMSPL and receives events that are then translated into new rules to be added in the local rules file. An example of SNORT3 rules is hereafter, an updated guide to rule writing is available here [14].

```plaintext
alert tcp $EXTERNAL_NET 80 -> $HOME_NET any
    (msg: "Attack attempt!");
    flow: to_client, established;
    file_data;
    content: "1337 hackz 1337", fast_pattern, nocase;
    service: http;
    sid: 1;
```
An example of data exchange is hereafter.

```json
{
    "type": "Mpolicy",
    "metadata fields...,
    "data": {
        "detc_rule": "...
    }
}
```

After adding a new signature to the detection rules set the *Engine* must be restarted.

### 4.2.5 Score Calculator

This sub-component will implement the *Device Trust Monitoring*, in the next version of the solution, it will provide a *monitoring score* based on the performance of a device in terms of packets generated with high priority: if the device generates lots of packets that the Engine detects as high priority threats then this sub-module create a Monitoring Score to be forwarded to the Broker through the MQTT channel (monitoringScore) triggering potential recalculation of the *Device Trust Score* by the Trust Manager.

<table>
<thead>
<tr>
<th>MQTT topic</th>
<th>Description</th>
<th>Module role</th>
</tr>
</thead>
<tbody>
<tr>
<td>monitoringScore</td>
<td>Topic for sharing data on high-priority alerts generators</td>
<td>Writer</td>
</tr>
<tr>
<td>DeviceData (Device/DID/newDevice)</td>
<td>Topic for sharing new device data (already bootstrapped)</td>
<td>Subscriber</td>
</tr>
</tbody>
</table>

Details on the definition of the Monitoring Score are not available yet. This module will be added in the next versions of the solution.

### 4.3 Interconnection with ERATOSTHENES tools

The three high-level activities mentioned in the introduction of chapter 4 expect the interconnection with other ERATOSTHENES modules.

In detail, the solution will cooperate with the following ERATOSTHENES modules:

- *CTI Agent* and the *MUD Management Module* for CTI Sharing

---

7 Topics names may change in next WP5 deliverables

© ERATOSTHENES, 2020
• Trust Manager for Device Trust Monitoring
• FedLPy to enhance ecosystem detection capability in general

4.3.1 Sequence diagrams

To better detail the high-level activities the following sequence diagrams depict the interaction required to perform them.

In Figure 6, the “DID Sharing” part is needed for the Device Trust Monitoring activity: when a new device is registered in the domain the IDS needs to store locally its IP address and the DID, and an initial score is attached to it, this score is then evaluated for any alert generated on its packets and edited in case of high priority. The devices that generate a lot of high-level alerts are notified to the Trust Manager which will request the recalculation of the Device Trust Score, “Monitoring Score sharing” part.

The two parts named “Threat sharing” and “Sec Policy sharing” are related to the CTI Sharing activity. In the first one, the alerts are notified to the CTI Agent, not all of the alerts but only the alerts with high-level priority. While, in both the update of the rules set is depicted but the request is provided by different entities (CTI Agent and MUD Management): the second interaction, “Sec Policy sharing”, is based on the measures shared by the MUD Management Module after, for example, a MUD certificate update which could require the adoption of new Engine’s rules.

The solution can also work as a mitigation actuator aid in Figure 7 the first part depicts a possible mitigation action: the IDS/IPS can block traffic from a specific device after receiving an untrusted DID - the IP of the device is added to the Engine blacklist.

The last part of the sequence diagram, in Figure 7, illustrates the request for device decommissioning: the IDS must remove from the list the IP/DID of the decommissioned device.

Figure 6 - Sequence diagram for IDS functionalities
For *Enhancing Detection capabilities*, the communication with the FedLPy is currently designed as follows:

➢ The ADI’s model is the initial FedLPy model to be forwarded to FedLPy edges – this way the edges will start the detection based on the feature generated by the ADI, and then incremental training will refine their detection capabilities, see Figure 8.

➢ The alerts generated by the FedLPy edges are collected by the IDS Threat/Rule Manager sub-module and displayed in the Alert GUI as part of the Monitoring module.

Implementation and other details will be reported in D4.7.

Another potential interconnection with the FedLPy is the identification of the IDS ADI sub-module as a FedLPy edge node, meaning the ADI could periodically receive an aggregated detection model provided by the FedLPy server and act as a FedLPy client, see details in *D4.4 Federated threat analysis models for continuous risk assessment*. This possible collaboration will be further studied with Task 4.4 partners for the next potential version of the solution.
4.4 Deployment

In Table 6, a complete list of dependencies, of the first version of the IDS/IPS solution is reported.

<table>
<thead>
<tr>
<th>Dependency</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNORT</td>
<td>3.1.57</td>
</tr>
<tr>
<td>Linux</td>
<td>Ubuntu 20.04</td>
</tr>
<tr>
<td>Python</td>
<td>3.9</td>
</tr>
<tr>
<td>MySQL</td>
<td>14.14</td>
</tr>
<tr>
<td>Nginx</td>
<td>1.18</td>
</tr>
<tr>
<td>Elastic stack</td>
<td>7.17.8</td>
</tr>
<tr>
<td>Node.js</td>
<td>12 or 14</td>
</tr>
<tr>
<td>MQTT.js</td>
<td>4.0.0</td>
</tr>
</tbody>
</table>

Initial list of Security Vulnerabilities has been provided as annex: it will provide the base to improve and the solution, also, from security point of view.
The Sub-components described in section 4.2 are implemented into a Docker container and must be deployed in a server-type machine with computational capability and high storage capacity, due to the performance provided by the *Engine* and the *ADI* sub-components.

The current version of the solution is provided as a Docker Image, stored in the project Nexus Docker Registry ([http://cysec-docker.eng.it/](http://cysec-docker.eng.it/)).

The image configures the *Engine*, starts the webserver for the *ADI* and the *Alert GUI* and starts the *Threat/Rule Manager* for CTI sharing.

The deployment is automated with the Docker-Compose tool: in Figure 9 the docker-compose defined for the solution deployment is depicted.

```yaml
services:

  bacon:
  
    container_name: BACON
    image: cysec-docker.eng.it/eratosthenes/ids-ips/engine:v1.2
    tty: true
    volumes:
      
        - type: volume
          source: log
          target: /var/log/snort
          
        - /bacon/usr/app/bacon
          
        - /snort-data/snort.lua:/usr/app/bacon/engine/snort.lua
          
        - /snort-data/local.rules:/usr/app/bacon/engine/local.rules
          
        - /bacon/start.py:/usr/app/bacon/start.py
          
        - /bacon/stop.py:/usr/app/bacon/stop.py
          
        - /bacon/requirements.txt:/usr/app/bacon/requirements.txt
          
        - /bacon/threattrulemanager/start.py:/usr/app/bacon/threattrulemanager/start.py
          
        - /bacon/threattrulemanager/stop.py:/usr/app/bacon/threattrulemanager/stop.py

  
    environment:
      
        - LS_JAVA_OPTS="-Xmx5120m-Xms5120m"
        - LS_HEAP_SIZE="5g"
        - discovery.type=single-mode
        - INTERFACE="wlp4s0"

    env_file:
      
        - .env

    network_mode: host

    depends_on:

    - tmdb
```

*Figure 9 - docker-compose.yml for the IDS/IPS solution*

To deploy the container just use `docker-compose up` command.
5 Research and Scientific Innovation

In this chapter, the partners involved describe the studies on what could be potential innovation ERATOSTHENES can bring to the detection capabilities already available in the market.

5.1 Anomaly Detection

Anomaly detection is a powerful technique for detecting deviations in data. Anomaly detection methods are required to enhance the effectiveness of a solution.

There are some inherent defects in traditional rule-based detection methods. For example, it is easy for attackers to bypass the predefined detection rules [15] and new unknown attacks cannot be discovered via the rules based on existing attacks. In essence, anomaly detection can be distinguishing abnormal data from normal behaviours.

To detect anomalies and interpret their meaning, one can use machine learning techniques.

In the context of the ERATHOSTENES project, anomaly detection can be achieved by:

- Defining anomaly base detection rules considering pilot’s requests.
- Using Machine Learning algorithms to analyse the IoT network traffic.

5.2 SOTA - Machine learning paradigms

Based on the state-of-the-art, there are three main paradigms of machine learning:

- **Supervised learning**: the training of the model is based on a dataset containing input and output data, or rather the *dataset* must be *labelled*. In supervised learning models the aim is basically to predict the labels without knowing them, based on the input data. Supervised learning can be used for different problems such as classifying if an email is spam or not.

- **Unsupervised learning**: there is not a direct relationship between the input and the output data. The *dataset is unlabelled*. In this case, the model determines correlations and relationships by analysing data. The main problem faced with unsupervised learning is clustering. Clustering involves grouping sets of similar data.

- **Reinforcement learning**: there is no input data to create a pattern from, and there is no output data to derive a relationship. Therefore, reinforcement learning uses an iterative approach to improving their decision-making, learning from their previous actions and experiences. Reinforcement learning is typically used in the fields of video games and robotics.

Previous studies on detection are usually connected to Supervised learning: classification has been used in most of the Intrusion Detection papers, the approach used in the ERATHOSTENES project, which could be the innovation brought, is to combine two machine learning algorithms, one unsupervised and then a supervised one to implement anomaly detection: this approach has been analysed and studied in other contexts but potentially could be applied in anomaly detection too [16] [17]. The idea is that by using a clustering algorithm on unlabelled datasets, before the classification one, the accuracy of the classifier is higher. The classification algorithms will be trained using the output data from the clustering algorithm.

The algorithms that can be used are different: based on the state-of-the-art, the most suitable supervised machine learning algorithms are Decision Tree, Naïve Bayes, Random Forest, and so. Based on the data that
the algorithm will be received, the best algorithm will be selected for pilots. While the unsupervised machine learning algorithms that can be used are DBSCAN, K-means, LOF, etc….9

The following diagram, Figure 10, depicts the high-level process described above, and details on the implemented procedure are reported in the next subchapter 5.3.

![Diagram of unsupervised and supervised learning algorithms]

5.3 Beyond SOTA

In the ERATOSTHENES project, the main idea is to utilize anomaly detection and signature-based detection, potentially integrated into a “zero-trust architecture”, never trust, always verify.

The IDS/IPS receives packets from the network. These packets will be analysed using anomaly detection techniques (implemented by the ADI) and signature-based techniques (implemented by the Engine). For each threat detected an alert is generated and stored locally.

The procedure defined to identify anomalies is shown in Figure 11: the upper part represents the Training, after which the ADI model is provided, and the bottom part is the Prediction, where the ADI model is applied to new records/packets to determine anomalies.

Specifically, they are divided into four phases:

- **Clustering phase**: uses an unsupervised algorithm. In this phase, \( K \) clusters will be created from a dataset that describes the normal network behaviour.
- **Outlier detection phase**: in this phase, all the outliers, from the \( K \) clusters, are removed from the dataset because they are considered noise and will be compromising the performance of algorithms. The resulting dataset will be combined with another known dataset (for example IoT23\(^{10}\) dataset or others).
- **Classification phase**: the classification algorithm will be trained using the combined dataset from the previous phase. The output of this phase is the ADI model, a trained machine-learning model which will be used in the last phase.
- **Predict phase**: in this phase, the new incoming traffic will be analysed and classified as normal or abnormal using the ADI model.

### 5.3.1 Testing Results

Initial studies will be carried out using two known datasets, **IoT-23**\(^{11}\) provided by Stratosphere Laboratory, AIC group, FEL, CTU University, Czech Republic and **NF-Ton-IoT**\(^{12}\) provided by Cyber Range and IoT Labs, the School of Engineering and Information Technology (SEIT), UNSW Canberra @ the Australian Defence Force Academy (ADFA).

The study was divided into two analyses, in both only the NF-TON-IoT and IoT-23 have been used (in the next studies pilots datasets will be added in the process as depicted in Figure 11):

1. the first using only classification algorithms
2. the second using the combination of the clustering algorithm and classification algorithms

---

\(^{10}\) [https://www.stratosphereips.org/datasets-iot23](https://www.stratosphereips.org/datasets-iot23)

\(^{11}\) [https://www.stratosphereips.org/datasets-iot23](https://www.stratosphereips.org/datasets-iot23)

\(^{12}\) [https://research.unsw.edu.au/projects/toniot-datasets](https://research.unsw.edu.au/projects/toniot-datasets)
As the clustering algorithm, the DBSCAN algorithm (Density-Based Spatial Clustering of Applications with Noise) has been selected, and for the classification phase, several algorithms are selected, and the results compared: Random Forest, Decision Tree, Support Vector Machine and Logistic Regression.

In this document are reported the results of the NF-ToN-IoT dataset and the initial analysis of the IoT-23 dataset.

### 5.3.1.1 Dataset description

NF-ToN-IoT includes telemetry data of Internet of Things (IoT) services, network traffic of IoT networks and operating system logs. The dataset consists of 22,339,021 rows in total between benign and attack.

The following list (Table 7) shows the attacks present in the dataset:

<table>
<thead>
<tr>
<th>Attack type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backdoor</td>
<td>It allows attackers to quietly get access to the system by bypassing the security protocols and trying to steal sensitive/personal information</td>
</tr>
<tr>
<td>Denial of service (DoS)</td>
<td>The main purpose of this attack is to shut down the targeted servers or machines, making them inaccessible to their users</td>
</tr>
<tr>
<td>Distributed Denial of service (DDoS)</td>
<td>These attacks occur when a single server is targeted by multiple servers with a DoS attack.</td>
</tr>
<tr>
<td>Injection attacks</td>
<td>An attacker tries to inject the code or malware into a program on a computer.</td>
</tr>
<tr>
<td>Man in the Middle (MITM)</td>
<td>The major objective of this attack is to steal the person’s personal information such as login credentials, etc.</td>
</tr>
<tr>
<td>Password attack</td>
<td>These types of attacks use to take advantage of your personal information by decoding password</td>
</tr>
<tr>
<td>Ransomware</td>
<td>It is a type of malware that encrypts the data of a user or any organization and then demands a ransom payment for the decryption key</td>
</tr>
<tr>
<td>Scanning attack</td>
<td>The attacker scans the devices to collect information such as IP address etc. of these devices before launching advanced attacks to sabotage their security.</td>
</tr>
<tr>
<td>Cross-site scripting (XSS)</td>
<td>XSS code is illegally injected into a web page or web application.</td>
</tr>
</tbody>
</table>

The dataset contains 14 features which are described in Table 8 below:
Table 8 - NF-ToN-IoT features

<table>
<thead>
<tr>
<th>ID</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IN_BYTES</td>
<td>Incoming number of bytes</td>
</tr>
<tr>
<td>2</td>
<td>OUT_BYTES</td>
<td>Outgoing number of bytes</td>
</tr>
<tr>
<td>3</td>
<td>IN_PKTS</td>
<td>Incoming number of packets</td>
</tr>
<tr>
<td>4</td>
<td>OUT_PKTS</td>
<td>Outgoing number of packets</td>
</tr>
<tr>
<td>5</td>
<td>FLOW_DURATION_MILLISECONDS</td>
<td>Flow duration in milliseconds</td>
</tr>
<tr>
<td>6</td>
<td>IPV4_SRC_ADDR</td>
<td>IPv4 source address</td>
</tr>
<tr>
<td>7</td>
<td>IPV4_DST_ADDR</td>
<td>IPv4 destination address</td>
</tr>
<tr>
<td>8</td>
<td>L4_SRC_PORT</td>
<td>IPv4 source port number</td>
</tr>
<tr>
<td>9</td>
<td>L4_DST_PORT</td>
<td>IPv4 destination port number</td>
</tr>
<tr>
<td>10</td>
<td>PROTOCOL</td>
<td>IP protocol identifier byte</td>
</tr>
<tr>
<td>11</td>
<td>TCP_FLAG</td>
<td>Cumulative of all TCP flags</td>
</tr>
<tr>
<td>12</td>
<td>L7PROTO</td>
<td>Layer 7 protocol (numeric)</td>
</tr>
<tr>
<td>13</td>
<td>LABEL</td>
<td>the type of capture, benign or malicious</td>
</tr>
<tr>
<td>14</td>
<td>ATTACK</td>
<td>Attack type</td>
</tr>
</tbody>
</table>

For our purpose, it has been chosen the 10% of the entire dataset and features with the following ID 5, 6, 7, 8, 9, 10, 11, 12, 13.

The IoT-23 dataset consists of network traffic from three different smart home IoT devices. These devices are Amazon Echo, Philips HUE and Somfy Door Lock. It consists of 23 captures, also called scenarios, of which 20 are malicious captures and 3 are benign captures.

The combined file, all 23 captures, contains a total of 1.444.674 records, and it has several attacks labelled as follows in Table 9.

Table 9 - Attack types in IoT-23 combined dataset

<table>
<thead>
<tr>
<th>Attack type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack</td>
<td>This label indicates that there was some type of attack from the infected device to another host. (e.g., brute force)</td>
</tr>
<tr>
<td>Control and Command (C&amp;C)</td>
<td>Indicates that the infected device was connected to a CC server</td>
</tr>
<tr>
<td>FileDownload</td>
<td>Indicates that a file is being downloaded to our infected device.</td>
</tr>
<tr>
<td>Mirai</td>
<td>The attack is performed by the Mirai bot network</td>
</tr>
</tbody>
</table>
The attack is performed by the Torii bot network, a more sophisticated version of the Mirai network.

**DDoS**
The infected device is performing a Distributed denial of service

**HeartBeat**
Indicates that packets sent on this connection are used to keep track of the infected host by the C&C server.

**PartOfAHorizontalPortScan**
Indicates that the connections are used to do a horizontal port scan to gather information to perform further attacks.

**Okiru**
The attack is performed by the Okiru bot network, a more sophisticated version of the Mirai network.

The IoT-23 dataset has 23 features, listed in Table 10.14 describes:

<table>
<thead>
<tr>
<th>ID</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ts</td>
<td>the time when the capture was done, expressed in Unix Time</td>
</tr>
<tr>
<td>2</td>
<td>Uid</td>
<td>the ID of the capture</td>
</tr>
<tr>
<td>3</td>
<td>id_orig.h</td>
<td>the IP address where the attack happened, either IPv4 or IPv6</td>
</tr>
<tr>
<td>4</td>
<td>id_orig.p</td>
<td>the port used by the responder</td>
</tr>
<tr>
<td>5</td>
<td>id_resp.h</td>
<td>the IP address of the device on which the capture happened</td>
</tr>
<tr>
<td>6</td>
<td>id_resp.p</td>
<td>the port used for the response from the device where the capture happened</td>
</tr>
<tr>
<td>7</td>
<td>proto</td>
<td>the network protocol used for the data package</td>
</tr>
<tr>
<td>8</td>
<td>service</td>
<td>the application protocol</td>
</tr>
<tr>
<td>9</td>
<td>duration</td>
<td>the amount of time data was traded between the device and the attacker</td>
</tr>
<tr>
<td>10</td>
<td>orig_bytes</td>
<td>the amount of data sent to the device</td>
</tr>
<tr>
<td>11</td>
<td>resp_bytes</td>
<td>the amount of data sent by the device</td>
</tr>
<tr>
<td>12</td>
<td>conn_state</td>
<td>the state of the connection</td>
</tr>
</tbody>
</table>

---

14 https://essay.utwente.nl/81979/1/Stoian_BA_EEMCS.pdf
© ERATOSTHENES, 2020
5.3.1.2 Test results with classification algorithms

In this section, the results of the tests will be reported using only the classification algorithms. The entire dataset is split into two parts: 70% of the data is utilized to train the ML algorithms, while 30% of the data was used to test ML algorithms.

Before training the machine learning algorithms, it is good practice to pre-process raw data. Two techniques were chosen for these tests: **LabelEncoder** and **StandardScaler**, both are part of the **scikit-learn** library (one of the most widely used Python libraries).

- **LabelEncoder** is a technique that is used to convert categorical values in a column (basically non-numerical values) into numerical ones\(^\text{16}\).
- **StandardScaler** is a technique that transforms a series of values into a standard normal distribution with a mean equal to zero and a standard deviation equal to one\(^\text{17}\).

<table>
<thead>
<tr>
<th></th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>local_orig</td>
<td>whether the connection originated locally</td>
</tr>
<tr>
<td>14</td>
<td>local_resp</td>
<td>whether the response originated locally</td>
</tr>
<tr>
<td>15</td>
<td>missed_bytes</td>
<td>number of missed bytes in a message</td>
</tr>
<tr>
<td>16</td>
<td>history</td>
<td>the history of the state of the connection</td>
</tr>
<tr>
<td>17</td>
<td>orig_pkts</td>
<td>number of packets being sent to the device</td>
</tr>
<tr>
<td>18</td>
<td>orig_ip_bytes</td>
<td>number of bytes being sent to the device</td>
</tr>
<tr>
<td>19</td>
<td>resp_pkts</td>
<td>number of packets being sent from the device</td>
</tr>
<tr>
<td>20</td>
<td>resp_ip_bytes</td>
<td>number of bytes being sent from the device</td>
</tr>
<tr>
<td>21</td>
<td>tunnel_parents</td>
<td>the id of the connection, if tunneled</td>
</tr>
<tr>
<td>22</td>
<td>label</td>
<td>the type of capture, benign or malicious</td>
</tr>
<tr>
<td>23</td>
<td>detailed_label</td>
<td>if the capture is malicious, the type of capture, as described above</td>
</tr>
</tbody>
</table>

\(^\text{15}\) https://scikit-learn.org/stable/index.html
As motioned before, several classification algorithms have been tested. As evaluation metrics were chosen accuracy score and confusion matrix, which gives a quick overview of the distribution of False Negative (FN), False Positive (FP), True Negative (TN), and True Positive (TP) values.

\[ TP = \text{true positive} \rightarrow \text{Classified as anomalous correctly} \]

\[ TN = \text{true negative} \rightarrow \text{Classified as normal correctly} \]

\[ FP = \text{false positive} \rightarrow \text{Wrongly classified as anomalous} \]

\[ FN = \text{false negative} \rightarrow \text{Wrongly classified as normal} \]

The accuracy score is defined as the total number of correctly predicted records over the total number of records:

\[ A = \frac{TP + TN}{Tot} \]

**NT-ToN-IoT dataset**

As a first test, the analyses carried out using the NT-ToN-IoT dataset will be reported hereafter.

- **Decision Tree:**
  - The model's accuracy is 0.9996381764091019.
  - Confusion matrix: \(TN=5995, FP=6, FN=3, TP=18870\)

- **Random Forest:**
  - The model's accuracy is 0.9996381764091019
  - Confusion matrix: \(TN=5999, FP=2, FN=2, TP=18871\)
**Support Vector Machine:**
- The model’s accuracy is 0.9992361501969929
- Confusion matrix: TN=5993, FP=8, FN=11, TP=18862

**Logistic Regression:**
- The model’s accuracy is 0.9859692851973949
- Confusion matrix: TN=5784, FP=217, FN=132, TP=18741
5.3.1.3 *Test results using combined algorithms.*

In this section, the results of the tests will be reported using both unsupervised and supervised machine learning algorithms.

As mentioned before, for the unsupervised algorithm was used DBSCAN. The algorithm has two hyperparameters to be set:

- \( n \) - the minimum number of points (a threshold) clustered together for a region to be considered dense
- \( \varepsilon \) - a distance measure that will be used to locate the points in the neighbourhood of any point.

For our tests was chosen \( n = 2D \), where \( D \) is the size of the dataset (in other words, the number of features) and \( \varepsilon = 0.5 \). The application of DBSCAN has identified 45 clusters and 831 outliers. After that, the outliers have been removed and the dataset has been divided into 70% to train algorithms and 30% to test algorithms. Also in this case the evaluation metrics used are accuracy score and confusion matrix.

The results using the *NF-ToN-IoT dataset* are:

- **Decision Tree:**
  - The model’s accuracy is \( 0.9996751269035533 \)
  - Confusion matrix: \( \text{TN}=5774, \text{FP}=3, \text{FN}=5, \text{TP}=18843 \)

  ![Decision Tree confusion matrix](image)

- **Random Forest:**
  - The model’s accuracy is \( 0.9996751269035533 \)
  - Confusion matrix: \( \text{TN}=5774, \text{FP}=2, \text{FN}=2, \text{TP}=18846 \)
D4.5. Intrusion detection for IoT-based context and networks

- **Support Vector Machine:**
  - The model’s accuracy is 0.9996345177664975
  - Confusion matrix: TN=5774, FP=3, FN=6, TP=18842

- **Logistic Regression:**
  - The model’s accuracy is 0.9917157360406091
  - Confusion matrix: TN=5583, FP=194, FN=10, TP=18838
Summary of the tests - NT-ToN-IoT

Comparing results with both procedures with classification algorithm and combination of clustering and classification, there is a slightly higher value of accuracy using combined algorithms, especially for the Logistic Regression classifier, see summary in Table 11.

<table>
<thead>
<tr>
<th>NF-ToN-IoT</th>
<th>Classification</th>
<th>Clustering + Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Decision Tree</strong></td>
<td>0.9996381764091019</td>
<td>0.9996751269035533</td>
</tr>
<tr>
<td><strong>Random Forest</strong></td>
<td>0.9996381764091019</td>
<td>0.9996751269035533</td>
</tr>
<tr>
<td><strong>Support Vector Machine</strong></td>
<td>0.9992361501969929</td>
<td>0.9996345177664975</td>
</tr>
<tr>
<td><strong>Logistic Regression</strong></td>
<td>0.9859692851973949</td>
<td>0.9917157360406091</td>
</tr>
</tbody>
</table>
6 Conclusions

In modern ecosystems, security must be applied and maintained in each procedure, activity and business, but the introduction and evolution of IoT devices are providing security experts and teams with new challenges: massive network data traffic, detection efficiency and real-time detection are just a few of them.

ERATOSTHENES try to also work on one of the NIST framework's key functions, the Detec, which is to "develop and implement the appropriate activities to identify the occurrence of a cybersecurity event."\(^{18}\) : an Intrusion Detection System is been designed, implemented and integrated with the ERATOSTHENES framework to detect cybersecurity events and notify as soon as possible.

This deliverable collects the details from the design to the implementation, and initial interaction with other modules, of the first version of the proposed solution: in total, three main versions will be realised, and the detail will be reported in the next deliverable D4.7 and D4.8.

The work reported on this document is closely connected with other tasks, in particular:

- all the tasks from WP1 for the
- task T1.2 for the requirements and the use-cases
- task T1.4 for the architecture,
- task T2.1 For the cooperation and integration with the TMB modules
- task T4.4 for the cooperation and integration with the FedLpy
- task T4.6 for the cooperation and integration with the device lifecycle security
- task T5.1 for the integration
- task T5.2, T5.3, T5.4, T5.6 for input and feedback from the pilots

First, the D1.2 and D1.3 output has been analysed to define and design functionalities based on initial requirements. Then they have been partly implemented for this first version of the solution:

- CTI Sharing
- Anomaly detection

For the CTI Sharing, the detection capability provided by the Engine has been improved by notifying CTI Agent and receiving threat signatures to be added to the detection rule set.

For Anomaly detection, an initial procedure has been defined and implemented as a FastAPI solution and tested with known datasets: NT-ToN-IoT.

In the next versions, the defined functionalities will be further refined or implemented, such as the Device Trust Monitoring, and additional integration activities are foreseen, such as with T4.4 output.

Furthermore, it should be clear that details reported in this document may potentially change in the following reports based on architectural decisions, integration aspects and demonstration feedback.

\(^{18}\) https://www.nist.gov/cyberframework/getting-started/quick-start-guide
7 References


## ANNEX I: Open-source Security Vulnerabilities

Table 12 - Known vulnerabilities

<table>
<thead>
<tr>
<th>Open-source Solution</th>
<th>Version</th>
<th>CVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastAPI</td>
<td>&lt;0.65</td>
<td><strong>CVE-2021-32677</strong></td>
</tr>
<tr>
<td>SNORT</td>
<td>3.1</td>
<td><strong>CVE-2021-40116</strong></td>
</tr>
<tr>
<td>Python</td>
<td>*</td>
<td><a href="https://www.cvedetails.com/vulnerability-list/vendor_id-10210/product_id-18230/Python-Python.html">https://www.cvedetails.com/vulnerability-list/vendor_id-10210/product_id-18230/Python-Python.html</a></td>
</tr>
<tr>
<td>Elasticsearch</td>
<td>*</td>
<td><a href="https://www.cvedetails.com/vulnerability-list/vendor_id-13554/Elasticsearch.html">https://www.cvedetails.com/vulnerability-list/vendor_id-13554/Elasticsearch.html</a></td>
</tr>
<tr>
<td>MQTT.js</td>
<td>&lt;2.15</td>
<td><strong>CVE-2017-10910</strong></td>
</tr>
</tbody>
</table>