Proposal No. 830929

Call H2020-SU-ICT-03-2018

Project start: 1 February 2019

Project duration: 42 months

Cyber Security for Europe

D3.14

Cooperation with Threat Intelligence Services for deploying adaptive honeypots

Document Identification	
Due date	30 October 2021
Submission date	29 October 2021
Revision	2.05

Related WP	WP3	Dissemination Level	PU
Lead Participant	CNR	Lead Author	Massimo Guarascio, Giuseppe Manco (CNR)
Contributing Beneficiaries	KUL, POLITO, C3P, ATOS, UNITN, DTU, UMU, UMA	Related Deliverables	D3.3, D3.12

Abstract: This deliverable describes the demonstrator that we have set up. The demonstrator is based on a conceptual platform, meant for gathering and managing threat information from different data sources. Basically, the demonstrator has the twofold objective of (i) improving the accuracy of Threat Intelligence Services (such as, e.g., Threat Detection Systems (TDSs)) in detecting and characterizing incoming attacks, and (ii) enabling the sharing of trusted, reliable and relevant threat information (e.g., discovered by TDSs or gathered by means of honeypots) among organizations and threat detection algorithms.

This document is issued within the CyberSec4Europe project. This project has received funding from the European Union's Horizon 2020 Programme under grant agreement no. 830929.

Executive Summary

Sharing threat events and indicators of compromise (IoCs) enables quick and crucial decision making relative to effective countermeasures against cyberattacks. The current information sharing platforms, however, do not allow easy communication and knowledge sharing among threat detection systems (in particular Intrusion Detection Systems) which exploit machine learning capabilities.

The aim of task 3.14 *Cooperation with Threat Intelligence Services for deploying adaptive honeypots* consists in designing and developing an information sharing and awareness platform and proof-of-concept demonstrator supporting the cooperation among such detection systems and other information awareness components (e.g., *honeypots*, *privacy preserving modules*, etc.). The demonstrator is backed by a distributed Threat Intelligence Platform (TIP) composed of several interconnected Malware Information Sharing Platform (MISP) instances, which enable communication with different Threat Detection Systems (TDS). Within this ecosystem, Threat Intelligence Services mutually benefit by sharing knowledge that allows them to refine and integrate the underlying capabilities.

To demonstrate the potential of the developed prototype, three relevant use cases concerning the cooperation of TDSs and other Cybersecurity tools have been set up. The main idea consists in highlighting how the cooperation can (i) improve the performances of the threat prevention and detection systems; and (ii) enable more robust threat intelligence by devising flexible strategies, methodologies and data formats.

In more details, the first scenario focuses on sharing CTI within and across communities. By sharing cyber threat information, other stakeholders or systems can leverage the shared information and collaborate to further analyse the data, increase the confidence in the shared intelligence, or to augment it with additional information such as the reputation and trustworthiness of the reporting entities.

The second scenario shows how enriching the information on detected threats with further details provided by different TDSs: several TDSs belonging to the cooperation network share data (e.g., anomaly scores, attack classifications, etc.) on detected cyberattacks. Moreover, new attack instances gathered via a honeynet and shared with the TIP are used as further training data for AI-Based TDSs so to improve their effectiveness.

Gathering relevant information on attack strategies and sharing precious IoCs is the main aim of the last scenario. Basically, it aims at demonstrating how internal information gathered by means of a pool of honeypots can be used in the context of the security of an infrastructure.

Document information

Contributors

Name	Partner
Davy Preuveneers	KUL
Giuseppe Manco	CNR
Massimo Guarascio	CNR
Nunziato Cassavia	CNR
Susana Gonzalez Zarzosa	ATOS
Esteban Armas	ATOS
João Resende	C3P
Andrea Atzeni	POLITO
Jorge Bernal Bernabe	UMU
Luís Antunes	C3P
Rolando Martins	C3P
Weizhi Meng	DTU
Alba Hita	UMU
Ivan Pashchenko	UNITN
Daniele Canavese	POLITO
Rodrigo Roman	UMA

Reviewers

Name	Partner
Elias Athanasopoulos	UCY
Abdelmalek Benzekri	UPS-IRIT

History

Version	Date	Authors	Comment
0.1	2020-10-27	Giuseppe Manco	1st Draft
0.2	2020-10-27	Massimo Guarascio	Update to Section 2
0.3	2021-05-23	Davy Preuveneers	Updates in many sections
0.4	2021-06-23	Various authors	Updates in many sections. Collected contributions by several partners
0.5	2021-06-24	Giuseppe Manco	Migration to Microsoft Office. Major restructuring and updates in many sections
0.6	2021-06-27	Davy Preuveneers	Updates in executive summary and introduction, as well as minor updates in section 2, and major updates in section 3 for scenario 1
0.7	2021-07-21	Alba Hita	Updates in section 3, and minor changes in section 1 and 2
0.8	2021-07-22	Daniele Canavese	Updates in subsections 2.3.1.4, 3.2.2.2 and 3.2.3.1

0.9	2021-07-30	Alba Hita	Minor changes section 3.1.2, and updates section 3.2.3
0.10	2021-07-30	Davy Preuveneers	Updates to sections 1.2, 1.3 and 4 and minor corrections throughout the document
0.11	2021-08-09	João Resende	Updates to sections 2.3 and 3.3
0.12	2021-09-06	Ivan Pashchenko	Updates to section 3.3
0.13	2021-09-09	Davy Preuveneers	Various updates and corrections throught the document
0.14	2021-09-13	Weizhi Meng	Updates to section 3.2.2.4
0.15	2021-09-14	Rodrigo Roman	Updates to section 3.2.2.5
0.16	2021-09-15	João Resende	Updates to section 4
1.0	2021-09-16	Giuseppe Manco, Massimo Guarascio	1st Final version.
2.0	2021-10-08	Giuseppe Manco, Massimo Guarascio	Minor changes in different sections
2.01	2021-10-11	Davy Preuveneers	Minor fixes after review
2.02	2021-10-11	João Resende	Updates to section 4
2.03	2021-10-12	Giuseppe Manco	Minor fixes
2.04	2021-10-16	Various authors	Updates in many sections. Collected contributions by several partners
2.05	2021-10-28	Ahad Niknia	Final check, preparation and submission process

Table of Contents

List of Figures

List of Tables

List of Acronyms

A	**AAA**	Authentication, Authorization and Accounting
	AI	Artificial Intelligence
	API	Application Programming Interface
	APT	Advanced Persistent Threat
C	**CP-ABE**	Ciphertext-Policy Attribute-Based Encryption
	CPE	Common Platform Enumaration
	CTI	Cyber Threat Intelligence
	CVE	Common Vulnerabilities and Exposures
D	**DB**	Database
	DDoS	Distributed Denial of Service
	DL	Deep Learning
	DNN	Deep Neural Network
	DoS	Denial of Service
E	**EBIDS**	Ensemble Based Intrusion Detection System
	ENIDS	Edge Network Intrusion Detection System
H	**HIDS**	Host Intrusion Detection System
I	**IdP**	Identity Provider
	IDS	Intrusion Detection System

	IoC	Indicator Of Compromise
	IPS	Intrusion Prevention System
J	**JSON**	JavaScript Object Notation
M	**MAPE**	Monitor-Analyze-Plan-Execute
	MISP	Malware Information Sharing Platform
	ML	Machine Learning
N	**NIDS**	Network Intrusion Detection System
O	**OPA**	Open Policy Agent
	OSINT	Open Source INTelligence
P	**PCAP**	Packet Capture
	PET	Privacy Enhancing Technologies
	PP-CTI	Privacy Preserving Cyber Threat Intelligence
R	**REST**	Representational State Transfer
	RoCE	Risk of Compromise Estimation
S	**SEO**	Secure Event Object
	SOC	Security Operations Center
	SQL	Structured Query Language
	SSH	Secure SHell
	SSI	Self-Sovereign Identity

	SSL	Secure Sockets Layer
T	**TATIS**	Trustworthy APIs for Threat Intelligence Sharing
	TCP	Transmission Control Protocol
	TDS	Threat Detection System
	TEE	Trusted Execution Environment
	TIE	Threat Intelligence IntEgrator
	TIP	Threat Intelligence Platform
	TPM	Trusted Platform Module
U	**UI**	User Interface
W	**WS-Security**	Web Service Security
Z	**ZMQ**	ZeroMQ

1 Introduction

1.1 Motivation

Cyber Threat Intelligence platforms are widely considered valuable tools for easing the management of threat information: these solutions allow organizations to easily handle the whole process of gathering, pre-processing, enriching, correlating, analysing, and sharing threat events and associated data (Johnson, et al. 2016). According to (Dandurand and Serrano 2013), the main requirements for a Threat Intelligence Platform (TIP) can be summarized into (*i*) providing services for information sharing, (*ii*) automatizing the process, (*iii*) enabling capabilities for collaborative threat data analysis.

Although TIP technologies can potentially bring important benefits to the organizations, developing a comprehensive Cyber Threat Intelligence platform able to handle information from different sources is difficult to achieve (Zibak and Simpson 2019). A tentative set of guidelines for establishing and participating in cyber threat information sharing relationships has been proposed in (Johnson, et al. 2016). The authors define information sharing goals for organizations, identify threat information sources and propose the rules for managing the publication and distribution of the threat information of an organization with external ones. However, privacy and trust of the shared information are just some examples of the open challenges in defining a fully operational platform. Moreover, the lack of standards and solid approaches resulted in different combinations of products and methodologies (sometimes erroneously) labelled as threat intelligence.

The situation is further exacerbated by some specific challenges:

- The quality of threat feeds and events is not guaranteed and there is a need for a reliable and automated threat analysis and mitigation. This is particularly problematic for threat detection systems (such as IDSs) based on Artificial Intelligence/Machine Learning techniques, which are typically affected by high false positive rates thus nullifying both their detection capabilities and the informative content of their detections.
- The event-based sharing philosophy of threat intelligence platforms does not match well with data driven and AI powered threat intelligence. In particular, there is no easy way for two machine learning algorithms to share, analyse and compare their findings within a standardized framework that can boost their attack detection and mitigation strategies.

1.2 Our Approach

Due to the lack of effective solutions for the above-mentioned issues, in this task we defined a comprehensive platform and proof-of-concept demonstrator for information sharing and awareness able to provide (privacy preserving) key information about the threats suffered by a monitored system (e.g., *a computer network*), that is particularly focused on the issues introduced in the previous section. The requirements of the architecture come from the task 3.4 and deliverable D3.3 "*Research challenges and requirements to manage digital evidence*" which surveys the main state-of-the-art approaches adopted in this area as well as the gap in tackling functional and non-functional requirements. Additionally, the document D3.3 lists relevant components, algorithms and software included in the prototype.

The demonstrator has the twofold objective of (i) improving the TDS' accuracy in detecting incoming attacks by exploiting threat information gathered from different sources (e.g., honeypots), and (ii) enabling the sharing of reliable and relevant threat information among organizations and threat detection algorithms in a confidential and privacy-preserving manner. The guiding principle is that TDSs can benefit each other mutually by sharing knowledge, since a threat feed shared by a TDS can be exploited by another one to improve its threat modelling strategies. At the same time, the same feed can be

enriched by enabling communication with other layers, such as honeypot pools, which can be exploited to further characterize the underlying threats and provide important insights about them.

The framework is based on an exchange protocol where information concerning threats is published on TIPs and made accessible to other actors, which can then exploit their capabilities to further characterize the feed and acquire it in their knowledge base.

As we shall see, this collaboration framework exhibits several advantages. On the one hand, it allows for improving the performances of the threat prevention and detection systems and minimizing the attack surface by strengthening the robustness of machine learning and deep learning models, making them more robust to new threats, false positives and lowering the time to threat detection. On the other hand, it enables more robust threat intelligence by allowing to better contextualize threat data and devise flexible strategies, methodologies, and data formats for collaborative threat intelligence, also by improving the notification mechanisms to appropriately notify relevant stakeholders having different needs for a contextualized interpretation of threat data.

1.3 Document Structure

We provide an overview of the demonstrator architecture and a short description for each asset composing it in section 2. Section 3 introduces the use cases and, for each of them, some demonstrations that involve assets cooperating on specific tasks relative to these use cases. A summary and outlook, including an overview of relevant pointers to online proof-of-concepts, videos and software repositories that collectively demonstrate the cooperation with threat intelligence services and honeypots are provided in section 4.

2 Reference Framework

In this section, we will briefly outline the goals and scope of the reference framework behind our proof-of-concept demonstrator. We will discuss the architecture, functionalities and assets used for the cooperation with threat intelligence services and the deployment of adaptive honeypots.

2.1 Goals and Scope

Our goal is the demonstration of a modular cybersecurity platform able to provide key information about the status of a monitored system (e.g., a computer network). The focus should be on different aspects such as received and in progress attacks, indicators of compromise (IoCs), etc. We also aim at exploiting the honeypot technology in order to identify new types of threats and provide important insights about them. The objective is twofold:

- On the one hand, the proposed platform and the combination of the underlying technologies should improve the performances of the threat prevention and detection systems and minimize the attack surface by strengthening the robustness of machine learning and deep learning models, making them more robust to new threats, false positives and lowering the time to threat detection.
- On the other hand, we aim at enabling more robust threat intelligence by allowing to better contextualize threat data and devise flexible strategies, methodologies and data formats for collaborative threat intelligence, also by improving the notification mechanisms to appropriately notify relevant stakeholders having different needs for a contextualized interpretation of threat data.

2.2 Architecture and Functionalities

Figure 1 illustrates the main architecture behind the demonstrator. There are essentially three components that cooperate: Honeypots, TIPs and TDSs. Within each component, we list some of the assets[1] discussed in D3.3, which represent the key modules upon which we will develop the demonstrations.

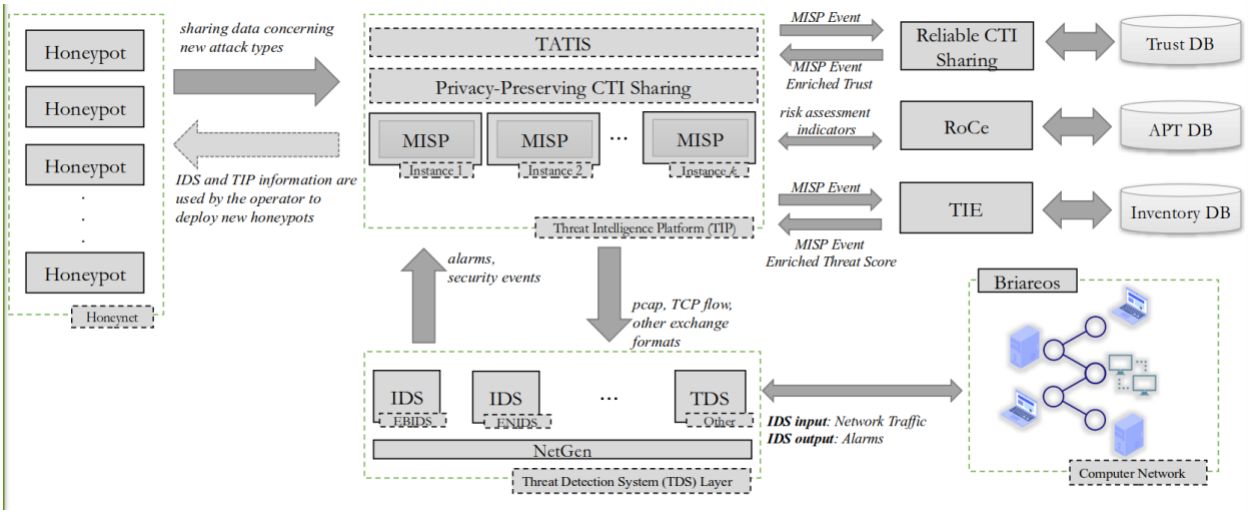

Figure 1 Overall Framework

[1] See section 2.3.1 for the details on such assets.

The **Threat Intelligence Platform** (**TIP**) represents the core component of the whole architecture. Basically, it has a twofold role: (*i*) storing data coming from heterogeneous sources in an encrypted and distributed way; and (*ii*) delivering the gathered information to the other components. In practice, several MISP instances can cooperate and share data about incoming events from different modules included in the platform. At this level, several assets can contribute:

- By protecting content with encryption and other privacy enhancing techniques (e.g., **TATIS**, **Privacy-Preserving CTI)** with the aim to share the gathered data among different organizations, such that only certain entities can decrypt the threat intelligence if the attributes used to construct their individual decryption key match the encryption policy of the ciphertext.
- By analyzing the MISP Events and, by adopting heuristic approaches, yielding enriched events to be stored in the MISP (e.g., **TIE**), or providing valuable risk assessment indicators (e.g., by interfacing with APT database, **RoCe**).

Threat detection and prevention systems can interface with the TIP by providing information concerning incoming attacks and feeding it with new intrusion events/statistics. We consider a TDS pool here, which once again can include several assets for real-time and off-line detection (e.g., **EBIDS**). In addition, IDSs can exploit information stored in the TIP to improve their models/rules/signatures. In principle, even information concerning specific IDS/IPS models can be shared through the TIP. Other assets with similar functionalities are **NetGen** and **ENIDS**.

Honeypots can be adaptively deployed to collect information concerning new attacks. The deployment can be performed manually by an operator based on the TIP information flow. A software component (e.g., the *Honeypot Log ETL* module in Scenario 2, see Figure 14) analyses Honeypot logs and uploads new intrusion detections to the TIP.

The demonstrator architecture of Figure 1 fits within the larger global architecture of WP3 as outlined and updated in the common framework of deliverable D3.12, as depicted in Figure 2.

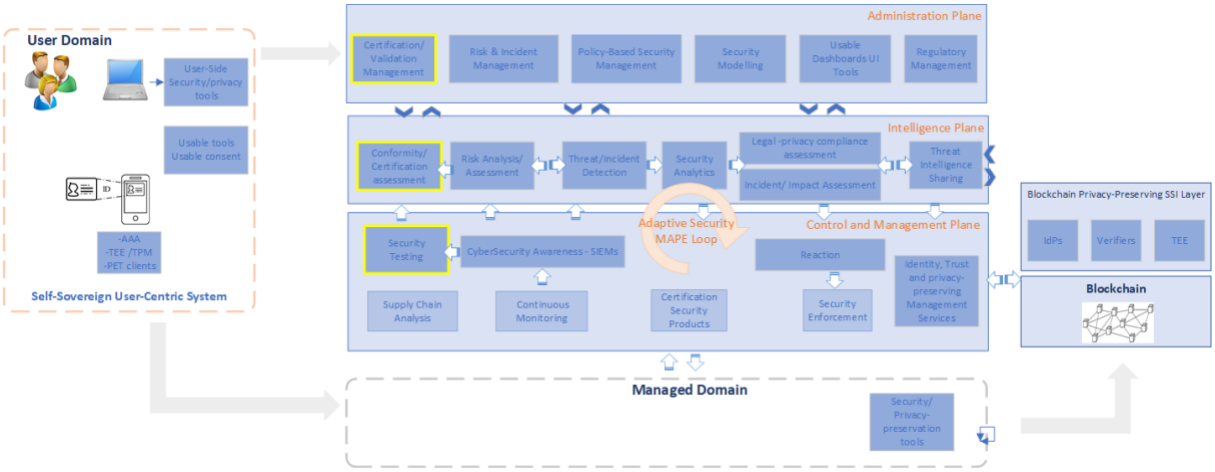

Figure 2 Global architecture of WP3

The demonstrator architecture and the assets used therein play a pivotal role in the 'Intelligence Plane' of the global architecture. In particular, it provides the capabilities of the following building blocks:

- Threat intelligence sharing
- Threat/incident detection
- Security analytics
- Incident/impact assessment

It also offers certain functionalities within the policy-based security management building block of the 'Administration Plane', for example by providing a policy-based approach to manage how threat intelligence information can be accessed and how sensitive information can be privatized.

2.3 Demonstrator Objects

The framework shown in Figure 1 requires essentially the cooperation of several assets that serve different purposes. In order to guarantee the cooperation, some additional artifacts need also to be taken into account. We will briefly introduce the assets and the additional components in the following subsections.

2.3.1 Assets

For each asset we provide a short summary description specifying the expected input and output with regards the proposed architecture. More details concerning the assets are available in deliverable D3.3.

2.3.1.1 TATIS [KUL]

TATIS (Preuveneers and Joosen 2020) (Preuveneers, Joosen and Bernal Bernabe, et al. 2020) adds a layer of abstraction to threat intelligence platforms, and MISP in particular, to guarantee the confidentiality of threat intelligence not only during transmission, but also when stored in the underlying database. As threat intelligence platforms share threat events, attributes and IoCs with other security stakeholders and organizations, TATIS encrypts threat events and attributes with CP-ABE, such that only authorized individuals with the right set of attestations can decrypt threat attributes. While initially developed as a reverse proxy for MISP, it can fulfil the same functionality for other threat intelligence and incident response solutions, such as The Hive[2].

Beyond CP-ABE, TATIS provides policy-based configuration capabilities to privatize or anonymize sensitive threat intelligence information before it is stored in MISP and possibly shared with other stakeholders or even other MISP instances. The enhanced confidentiality and privacy can be configured at the level of MISP events and individual attributes.

Last but not least, TATIS offers enhanced access control towards MISP's REST APIs. Traditionally, the latter are protected with a simple authorization key associated with a user in the system. TATIS augments this security with OAuth 2.0-based authorization and policy-based access control towards the APIs. To that end, TATIS leverages well-known open-source security building blocks, including Keycloak[3] for identity and access management, and OPA[4] for policy-based access control to the REST APIs.

2.3.1.2 EBIDS [CNR]

As described in D3.3, EBIDS (Folino, et al. 2021) is a Machine Learning based Intrusion Detection System which adopts specialized ensembles of classification models to identify undetected attacks by analysing network logs. With respect to the refence framework in Figure 1, EBIDS requires as input one or more packet capture (pcap) files to compute several statistics concerning the network traffic flow. These data are used as features during the training phase of the classification model. The ensemble classifier built by EBIDS can be regarded as an overall neural network that integrates two kinds of components: a number of base classifiers sharing the same architecture (but extracted from different

[2] https://thehive-project.org
[3] https://www.keycloak.org
[4] https://www.openpolicyagent.org

data chunks), and a combiner sub-net for deriving an overall prediction from the outputs of the base classifiers.

Specifically, the framework first builds a number of specialized bases DNN classifiers, induced from disjoint segments of a data stream, and next fuses the outputs of these classifiers by means of different combiner sub-nets. Then, the final output is an anomaly score (ranging in [0,1]) on the analysed traffic flow. The final ensemble model allows for discovering anomalous behaviours (potentially related with different types of cyber-attacks) which will be shared with the other software modules and organizations under the form of security events.

2.3.1.3 TIE [ATOS]

TIE is an automated system that evaluates and quantifies the impact of a security event within an organization's infrastructure. The security events generated and shared between interconnected institutions as well as external indicators of compromise received from open-source intelligence (OSINT) data sources facilitate the task of security teams when responding to possible attacks on their systems. However, these events may have little or great importance to one entity or another, depending on whether or not the event is related to a component within the entity. Therefore, an individual assessment of these security events is necessary. TIE facilitates the task of the security teams since it automatically evaluates and qualifies each of the incoming events. TIE evaluates the events considering the particular infrastructure of the executing entity and adjusts the event score in relation to the relevance for the entity.

To carry out its work, TIE receives events that arrive at the MISP instances of the connected entities. Subsequently, these events are analysed using the TIE's heuristic engine against events received in the past and an inventory database with information about the monitored infrastructure. Within the heuristic engine, each attribute and object that accompanies the event is evaluated considering five characteristics: Timeliness, Variety, Completeness, Accuracy, and Relevance. Each of the above characteristics can have a value between 0 and 5, values that are assigned based on the expert knowledge and utility criteria identified for the system. As a result, an enriched event with the final score calculated is shared to the MISP instance.

2.3.1.4 NetGen [POLITO]

NetGen is an Intrusion Detection System (IDS) generator that simplifies the creation of IDSs based on machine and deep learning techniques. To produce a new IDS, NetGen requires a collection of capture files (in pcap format) acting as the training set for training a set of supervised machine learning models. These files will be used to generate several different models and NetGen will automatically select the best one via a (customizable) Bayesian optimization process. Once the best IDS has been produced by NetGen, it can be quickly deployed for sniffing and analysing the traffic on a network interface or integrated in an existing network using its capabilities as a Python package.

Under the hood, NetGen analyzes the raw bytes of the traffic, reconstructs the TCP flows and, for each connection, computes various statistics (e.g., the number of transmitted bytes, the number of packets with some specific flags sets) and uses this data as the input features for a variety of supervised machine learning models (e.g., random forests, fully connected and recurrent neural networks). All the traffic statistics are computed by analysing only the TCP header, avoiding any kind of deep packet inspection. This makes the NetGen approach encryption-agnostic, being suitable also for secure connections (e.g., via SSL, SSH or WS-Security) and when the privacy of users is at stake.

NetGen is flexible enough to be used for detecting a wide variety of network attacks. Our experiments have shown that it can be successfully used to detect with high accuracy crypto-mining flows, SQL injections, DoS attacks, vulnerability scans and various kinds of web attacks.

2.3.1.5 Briareos [C3P]

Briareos is a modular framework designed for improving network security by detecting and preventing intrusions. It achieves this by extending rule-based Network Intrusion Detection Systems (NIDS), which

detect attack vectors with a known signature (i.e., already reflected in existing rules), with the capability of inferring new rules for unknown attack vectors (not covered by existing rules) by inspecting network traffic (both inbound and outbound) and correlating it with operating system behaviour. Its modular architecture allows Briareos to provide users with the possibility of building new modules and creating new processing pipelines for handling intricate intrusion schemes, not possible by traditional rule-based systems. In essence, Briareos allows networks and systems to be increasingly secure over time by detecting and classifying unknown attacks and preventing the system from future attack occurrences. Focusing on performance, Briareos offers adjustable security levels that provide system administrators a tradeoff between performance and the level of security intended for a host or a set of hosts. Briareos also supports the deployment on the devices of a small footprint Honeypot. This Honeypot allows detecting attacks on the internal network, creating the possibility of analysing events and scans of attackers based on known CVE.

2.3.1.6 Privacy-Preserving CTI and Reliable CTI Sharing [UMU]

Privacy-Preserving CTI Sharing is intended to provide anonymization over CTI events shared through TIPs, especially MISP. Towards this end, this asset enables the application of different PET mechanisms and anonymization methods, such as k-anonymity or generalization, which allow to obfuscate certain parts of information that could compromise privacy of stakeholders in case of disclosure. To provide its functionality, PP-CTI expects JSON-formatted MISP events as input, by following data taxonomies and models previously defined. Then, based on such taxonomies and models, this asset executes the corresponding PET mechanism and returns a new anonymized CTI event, which can be further shared with other stakeholders while privacy is still preserved.

Moreover, Reliable CTI Sharing enables an analysis of CTI-related data, which are shared through TIPs. This analysis not only provides an evaluation of the trustworthiness of these exchanged data, but also the organizations, sources and the relationships involved in the sharing process. Thus, Reliable CTI is designed to act as a service of TIPs, in particular, MISP instances. Therefore, inputs expected by this asset are JSON-encoded CTI events or information synchronized from MISP with ZeroMQ or via API REST. Regarding the output, it is an enriched CTI event with the computed value of reliability.

2.3.1.7 IntelFrame [DTU]

As data (e.g., threat events) is continuously shared within the threat intelligence platforms, there is a need to intelligently handle the increasing data and identify anomalies. IntelFrame aims to provide a method of selecting more suitable learning classifiers based on different contexts, or selecting a suitable parameter / setting for a particular classifier such as deep learning. The classifier can be used for reducing false positives and directly for improving detection performance, according to a concrete scenario. When the environment becomes large and distributed, the merit is that the classifier could be adaptive and distinct for different nodes.

2.3.1.8 RoCe [UNITN]

Risk of Compromise estimation (RoCe) is an experimental methodology that estimates the empirical hardness of exploiting the vulnerabilities in a network. Our key idea is to monitor components used in the network, in particular, upgrades of the software projects installed on the network components. Then we could compare the components used with the historical information of known advanced persistent threats (APT) and attacks. Combined, this information allows us to build a model that predicts the risk of compromise of a given network segment.

2.3.1.9 HADES [UMA]

In case a strand of malware is captured during the detection process, it is necessary to make use of data enrichment platforms to automatically extract information about such malware and share it with other platform modules. Precisely, HADES – introduced in deliverable D3.3 – is a tool that orchestrates

sandboxes for malware execution. In particular, it integrates the cuckoo sandbox to provide information like API calls and advanced memory analysis. As for its functionality, HADES can send any input samples to virtual machines, execute them, analyze their behaviour, and create output reports based on the proof generated. All this process is performed automatically, which facilitates its integration with any external platform.

2.3.2 Additional components

2.3.2.1 TDS data exchange format

Within the proposed framework, assets interact by exploiting MISP events. It is possible to define customization for such events by devising custom MISP Objects in JSON format. A MISP object represents the data structure adopted by MISP to store shared threat events. The general template can be extended to include further relevant information on specific threat events. Here we devise a custom MISP template, which can embed the whole set of data of relevance for a TDS. These include:

- Access to reliable threats, that is, threats for which a consensus and interpretation can be achieved. The available information can then be used to improve the detection capabilities. For traditional IDS like *Suricata*[5] or *Snort*[6] this is achieved by devising new security policy rules. However, data-aware IDS should be able to exploit such information to extend their training sets and rebuild the underlying ML models with improved accuracy.
- Export of the discovered threats to the MISP network, for further investigation and confirmation analysis. In particular, the information concerning the threats should allow further labelling/categorization by other actors/TDS accessing it.

Object attribute	MISP Attribute type	Description	Disable correlation	Multiple
creation-date	datetime	Threat Event Date	✓	-
ip_dst	ip-dst	Destination IP	-	✓
ip_dist_port	port	Destination Port	-	✓
ip_src	ip-src	Source IP	-	✓
pcap_file	attachment	PCAP File	✓	
verified	boolean	It specifies if an operator verified the occurrence of the attack	✓	-
signature_type	text	Type of signature (md5, sha1, …)	✓	-
signature	text	Optional detected file signature	-	✓
attack_type	text	A JSON containing information on IDS classification	-	-

[5] https://suricata.readthedocs.io/en/latest/lua/index.html

[6] https://www.snort.org/

anomaly_details	attachment	Optional JSON file containing anomaly flow statistics	-	-
privatized	attachment	Privatized version of the attributes	✓	✓

Table 1 Custom MISP Object fields

The components of the Custom MISP object are illustrated in Table 1. Some of them are particularly important:

- `pcap_file`: This file contains the (anomalous) network activity between two devices in ".pcap" Wireshark format. It is essentially the information exploited by the IDS to detect the threat and as such it is fundamental for the purposes of the sharing philosophy of the platform. In fact, it is the main piece of information to analyse for confirming the threat. Also, in case a confirmation is available, it can be exploited for enriching the knowledge-base of all the TDS accessing it. Since it can potentially contain sensitive information, it can be encrypted for secure sharing with only trusted sources.
- `attack_type`: This field stores, in json format, relevant information about the threat. It essentially includes a list of classification events relative to the pcap associated with the MISP object. Each event can be relative to a specific IDS and multiple IDS can contribute with entries within `attack_type`. For example, if algorithm *X* detected a DOS attack within the pcap file, then `attack_type` includes an entry containing the attack label (DOS), a confidence value/anomaly score, and the signature of the detection algorithm.
- `anomaly_details`: An optional JSON fragment containing aggregate statistics and further information exploited by the IDS for characterizing the threat. For example, it can include some summary statistics extracted from the pcap file, or the signatures used to characterize it.

Figure 3 shows an example instance. Here, `attack_type` contains JSON instantiated with a specific entry, in lines 32-39. It refers to a specific ML-Based IDS (i.e., EBIDS), which relatively to the attached pcap file (`anomaly_pcap`, as declared in the `pcap_file` attribute on lines 40-44), detects an anomalous traffic flow (labelled as `ANOMALY`) with a 98% confidence level.

```
1.  {"object": [{
2.          "id": "18919",
3.          "name": "security_event_object",
4.          "description": "CS4E Security Event Object",
5.          "uuid": "ba14028e-03a6-47d2-b35f-8147381493cf",
6.          "timestamp": "1615454674",
7.          "Attribute": [{
8.              "id": "262154",
9.              "type": "ip-src",
10.             "category": "Network activity",
11.             "object_relation": "ip_src",
12.             "value": ""},{
13.             "id": "262155",
14.             "type": "ip-dst",
15.             "category": "Network activity",
16.             "object_relation": "ip_dst",
17.             "value": ""},{
18.             "id": "262156",
19.             "type": "port",
20.             "category": "Network activity",
21.             "object_relation": "ip dst port",
22.             "value": ""},{
23.             "id": "262157",
24.             "type": "datetime",
```

```
            "category": "Other",
            "object_relation": "creation-date",
            "value": "2021-03-11T18:24:34.194148+0000"},{
            "id": "262158",
            "type": "text",
            "category": "Other",
            "object_relation": "attack_type",
            "value": {
              "EBIDS": {
                "version": "0.2",
                "reference": "https://github.com/ebids/",
                "attacks": [{
                    "attack_type": "ANOMALY",
                    "confidence": "0.98153543"
                  }]}}},{
            "id": "262159",
            "type": "attachment",
            "category": "External analysis",
            "object_relation": "pcap_file",
            "value": "anomaly.pcap"},{
            "id": "262160",
            "type": "attachment",
            "category": "External analysis",
            "object_relation": "anomaly_details",
            "value": "anomaly.json"},{
            "id": "262161",
            "type": "boolean",
            "category": "Other",
            "object_relation": "verified",
            "value": "0"
        }]}]}
```


Figure 3 Security Event Objects

3 Use Cases

In the following we list three complementary scenarios that we use to demonstrate the collaboration framework devised in the previous section. For each scenario, we describe the intended purpose and the assets involved. Next, specific demonstrations illustrate how the scenario is realized.

3.1 Scenario 1. Sharing CTI

3.1.1 Summary

The focus of the first scenario is on the sharing of cyber threat intelligence (CTI) within and across communities, as this is a key enabler for cooperation between threat intelligence services and to trigger the deployment of adaptive honeypots. By sharing cyber threat information, other stakeholders or systems can leverage the shared information and collaborate to further analyse the data, increase the confidence in the shared intelligence, or to augment it with additional information such as the reputation and trustworthiness of the reporting entities. Additionally, the information can be leveraged in an adaptive honeypot that aims to deploy software vulnerable against reported threats as a way to lure adversaries and gather more intelligence while they attack the reported vulnerabilities. The sharing capability within and across communities is already present within state-of-practice cyber threat intelligence platforms, though due to the sensitive nature targets of cyberthreats are not always willing to share this information unless they are obliged to be compliant with mandatory incident reporting regulations.

This scenario demonstrates how we address this concern. As we are dealing with sensitive information about threat targets and detailed information about vulnerabilities, the information should never end up in the wrong hands, e.g., malicious adversaries that aim to exploit the intelligence. This scenario builds upon state-of-practice and open-source threat intelligence tools for storing and sharing information to further strengthen the security and privacy posture of intelligence sharing. The main objectives of this scenario are to mitigate privacy concerns by (1) pre-processing threat intelligence with well-known privacy enhancing technologies and data anonymization techniques before the intelligence is shared, and (2) cryptographically protecting the shared threat intelligence in a fine-grained manner so that only authorized entities can decrypt and act upon it.

The following scenarios about enriching CTI and adaptive honeypots can leverage the capabilities offered by this scenario to request access to privatized and protected information, or to protect their own information before sharing within and across the community.

3.1.2 Assets Roles

3.1.2.1 TATIS

TATIS plays a role in maintaining the confidentiality and privacy of threat events at a more fine-grained level than what the typical Traffic Light Protocol is able to when synchronizing threat events from one MISP platform deployed to the other. TATIS can be used to encrypt or privatize network related threat information provided by other NIDS or HIDS solutions, using CP-ABE or other anonymization techniques for a more trustworthy sharing of threat intelligence. For the sharing of threat intelligence, multiple MISP instances belonging to different project partners have been set up and interconnected with one another.

KU Leuven has also set up a standalone MISP environment that is interconnected with The Hive 4 security incident response platform, as well as a Cortex 3 instance with a variety of observable analyzers and operation automation responders. This way, published MISP threat events encrypted with CP-ABE through TATIS are shared with The Hive platform in an encrypted form. Custom Cortex analyzers and responders interact with TATIS to decrypt the threat intelligence and take appropriate action. This may include enriching the encrypted threat intelligence, as well as acting upon the threats by invoking or reconfiguring other assets, such as a honeypot.

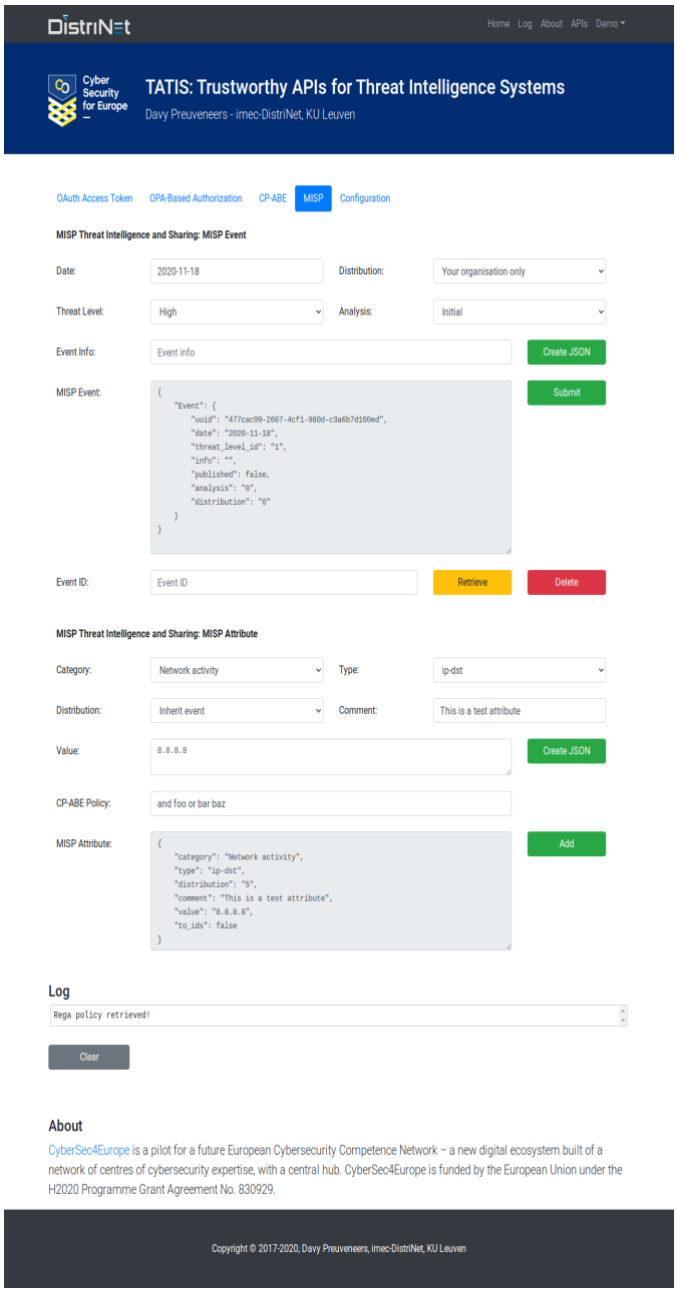

Figure 4 Online instance of the TATIS asset

Figure 4 illustrates an online instance of TATIS interacting with a locally deployed MISP instance to CP-ABE encrypt and decrypt events and attributes. TATIS can also be configured to interact with MISP instances offered by other parties. The HTML5 user interface invokes APIs exposed by TATIS to store and retrieve encrypted and/or privatized threat intelligence in MISP. The same APIs can be used by other assets, both for ingress and egress. To make use of TATIS APIs, an OAuth bearer token needs to

be provided. This functionality is made available by a Keycloak instance. The TATIS APIs are protected by a RedHat Keycloak 11.0 instance by means of policy-based access control using Open Policy Agent as an externalized policy engine. The latter is mainly meant as a proof-of-concept to demonstrate the more fine-grained and flexible API security compared to what MISP, for example, offers out of the box with their fixed authorization keys. More technical details are available in (Preuveneers and Joosen 2020).

Next to the HTML5 interface, TATIS also exposes its functionality for testing purposes through an online OpenAPI 3.0 specification. Note that it is also possible to directly interact with the MISP instance, or indirectly through TATIS without relying on CP-ABE cryptography or any other privacy enhancing technique. To do so, TATIS provides a policy-based approach to indicate how individual events and attributes should be processed to enhance confidentiality or privacy. The listing below offers a subset of the policy used in the demonstrator.

```
{
  "attributes": [
    {
      "name": "ip-dst",
      "pets": [
        {
          "scheme": "cpabe",
          "metadata": {
            "policy": "and foo or bar baz"
          }
        },
        {
          "scheme": "sha256"
        },
        {
          "scheme": "pbkdf2",
          "metadata": {
            "iterations": 1000
          }
        }
      ]
    }
    ...
  ]
}
```

Figure 5 Example privacy policy of TATIS

The advantage of this approach is that it can be easily enhanced with new schemes and additional metadata.

Finally, TATIS was designed in such a way that the other assets, including those detecting threats, can leverage existing frameworks such as the PyMISP[7] library to report their threat intelligence through TATIS to MISP.

3.1.2.2 Privacy-Preserving CTI Sharing asset

The Privacy-Preserving CTI Sharing asset aims to carry out an anonymization process over sensitive information from threat events by applying PET mechanisms. This component can be used to apply PET mechanisms, such as pseudonymization or k-anonymity, to the IoCs and attributes of a threat event. This way, the information shared is obfuscated and can be made harmless if any leakage occurs.

These privacy-enhancing techniques could be complimentary to the cryptographic techniques, such as CP-ABE, that TATIS applies. The difference lies in the fact that PET techniques applied with this asset

[7] https://www.circl.lu/doc/misp/pymisp/

anonymise a dataset by making it impossible to access the raw data, whereas the techniques applied by TATIS protect access to the raw data so that only the relevant individuals can access the raw data.

This component is designed as an external service to the TIPs providing a REST API to interact with. Or it can be an external complementary service of TATIS, for this demonstration. Therefore, when a NIDS or HIDS reports a threat event to TATIS, this asset could forward the event with some privacy policies to the PP-CTI Sharing service. According to those policies the defined data taxonomies and models, the Privacy-Preserving CTI Sharing can apply anonymization techniques to obfuscate a part of the information included in the events. Finally, PP-CTI Sharing asset sends back a new anonymized threat event to TATIS, which acts as a reverse proxy and forwards it to MISP for its sharing with other stakeholders and organizations while privacy is preserved.

3.1.3 Demonstrations

The two main assets in this scenario are complementary in nature. While TATIS already offers several anonymization/pseudonymization tactics and privacy enhancing techniques, it can also make use of capabilities offered by other parties, such as PP-CTI. TATIS implements a customizable policy that defines how attributes and events should be privatized and protected. Within this policy, the use of external services can be configured. By having TATIS acting as an abstraction layer, the advantage for the assets in the other scenarios is that they do not have interact with different privacy enhancing building blocks, nor do they need to update their own implementation logic when other privacy enhancing capabilities are directly added to TATIS or offered by a new external service.

An online instance of TATIS is available at:

- https://ciel.cs.kuleuven.be/tatis/

A video demonstration of the TATIS asset in action is available at:

- https://people.cs.kuleuven.be/~davy.preuveneers/tatis.mp4

Authorization policy for MISP REST APIs

TATIS operates as a reverse proxy for MISP. The latter offers REST APIs which are protected with a static authentication key. TATIS enhances the access control to these APIs by implementing policy-based access control using the OPA policy engine. This way, access to the APIs can be restricted with attribute-based access control in a more fine-grained manner than what MISP is currently capable of. The same protection mechanism of TATIS can also be used to protect other CTI tools that offer REST APIs to provide their services.

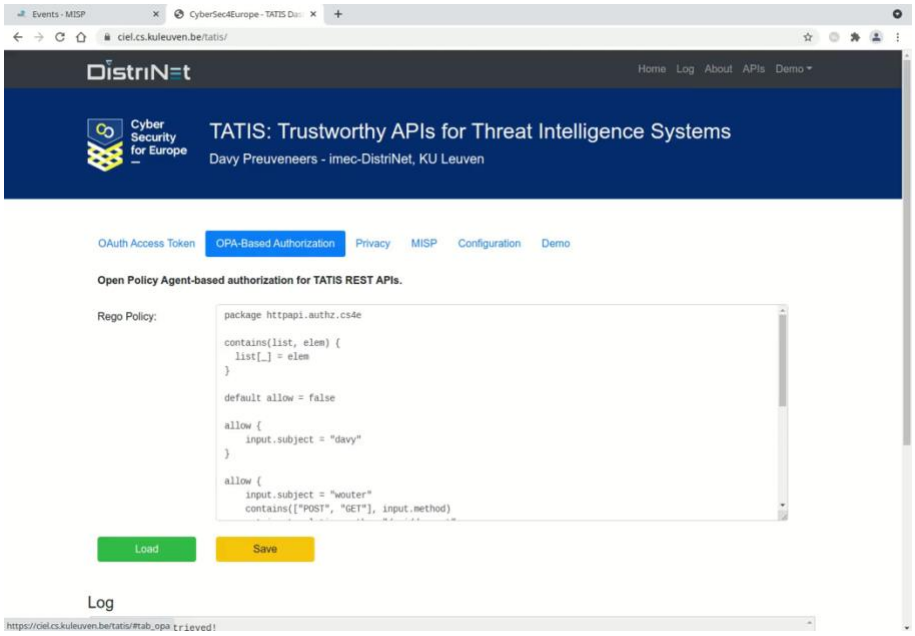

Figure 6 Attribute based access control to TATIS's REST APIs using a Rego policy and the Open Policy Agent (OPA) engine

Privacy policy for MISP attributes

As depicted in the figure below, TATIS implements a JSON based policy to indicate how different attributes in a threat event should be pre-processed before it is forwarded and stored in MISP. The example below depicts how the "ip-dst" attribute is protected with CP-ABE using the 'and foo or bar baz' policy (such that only those entities that match this particular policy can decrypt the information). Additionally, the same attribute is offered in a hashed form using SHA256. If a user does not match the CP-ABE policy, it can still use the hashed value for correlation analysis. The last example is only for demonstration purposes and not really a secure privacy enhancing tactic, as a curious recipient may reveal the original IP address by launching a rainbow table attack. However, for longer and more random input fields, cryptographic hashing might be a useful tactic without revealing the raw sensitive information.

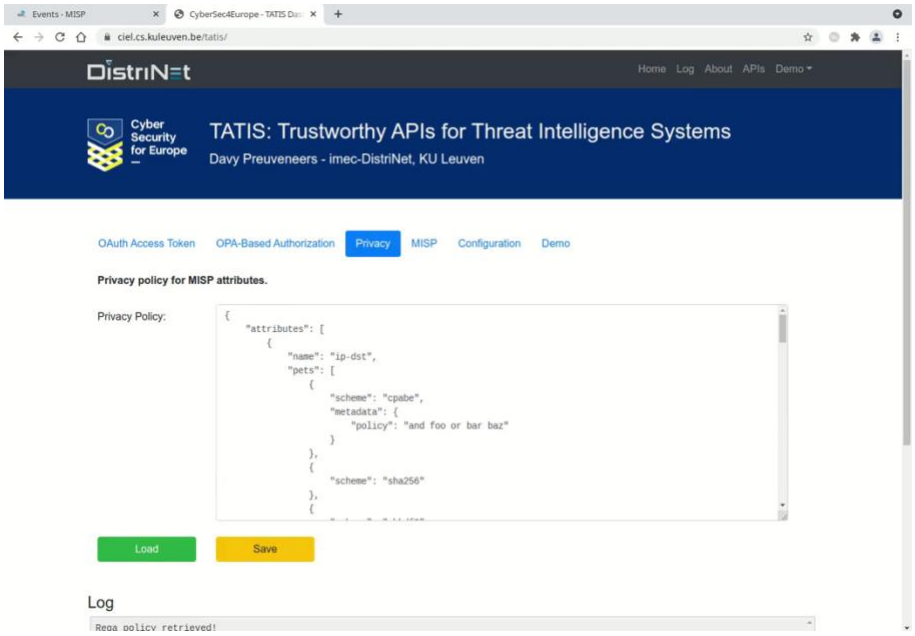

Figure 7 A JSON privacy policy to protect the confidentiality and privacy of MISP attributes

Protecting MISP attributes

The TATIS dashboard can be used to protect a sensitive attribute using the privacy policy depicted above. The outcome is that the confidentiality of the MISP attribute (in this case `ip-dst`, the IP destination address) is enhanced with a.o. CP-ABE (only authorized parties can retrieve the plaintext IP address) and SHA256 (for all other parties). TATIS transforms the attribute by converting its type into a MISP attachment, and storing the privatized information within a zip file, in this case with the random filename `4c39cf4e102d4ce6aef962af80890546.zip`. The figure below depicts the revised JSON representation of the MISP attribute before it is stored in MISP's backend.

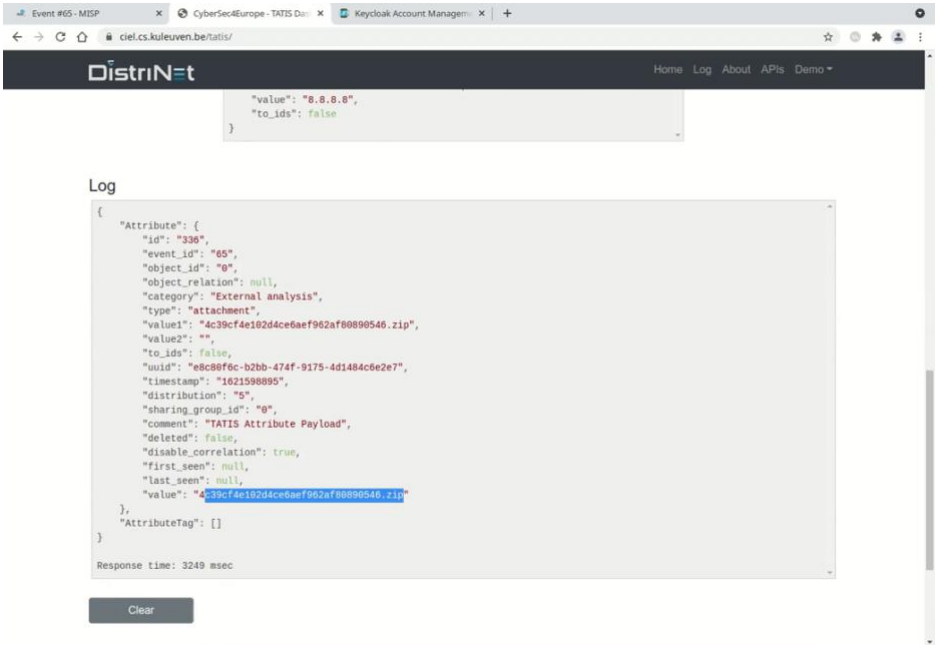

Figure 8 Using TATIS to privatize a sensitive MISP attribute into a ZIP archive containing the protected threat intelligence.

The privatized and protected threat intelligence is shared as a ZIP archive in MISP, as can be seen through MISP's dashboard in the figure below. Note that the security of our tool is not based on the ability to protect ZIP files with a password, but the fact that the files in the ZIP archive themselves are protected. The rationale for using a ZIP archive is to privatize the same MISP attribute in different manners such that the recipient may use different transformations of the same attribute depending on its access and the trust level.

Any tool that interacts with MISP can retrieve the ZIP file, and use TATIS to decrypt its content for further offline analysis, etc.

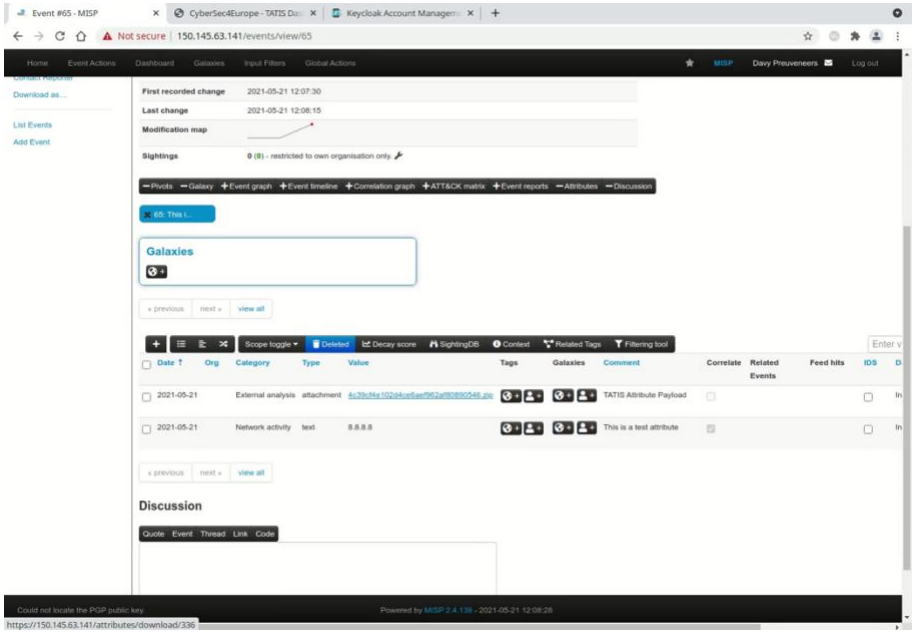

Figure 9 Retrieving the same ZIP archive through the MISP dashboard.

Retrieving and decrypting protected MISP attributes

Alternatively, one can use the TATIS dashboard to retrieve the CP-ABE decryption key as depicted in the figure below. A user only obtains this private key whenever the user's personal attributes and claims stored in the Identity and Access Management platform (our scenario uses Keycloak for this purpose) match the CP-ABE policy with which the attribute is encrypted.

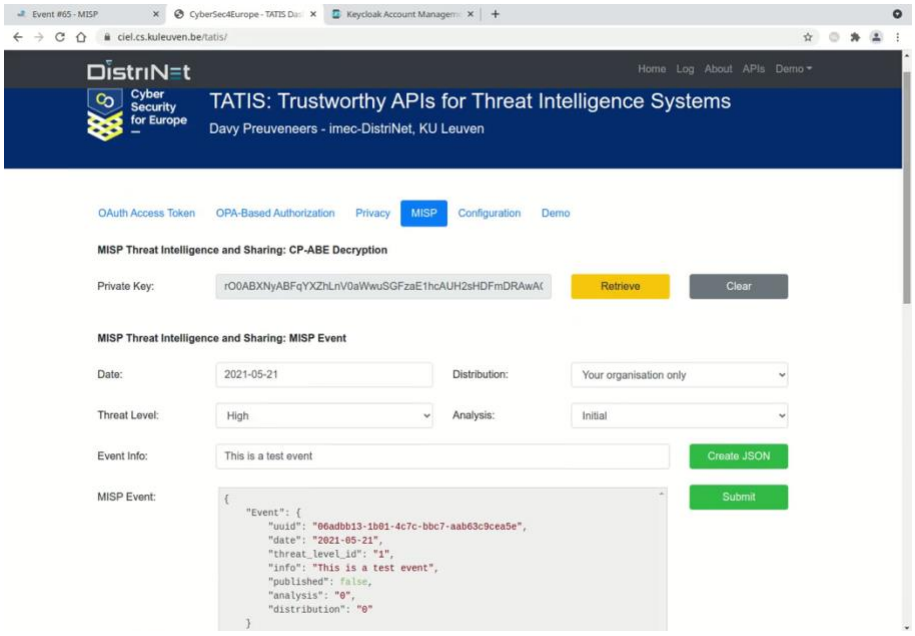

Figure 10 Obtaining the CP-ABE decryption key through the TATIS dashboard.

After receiving the private decryption key, and requesting the same MISP event, the protected attribute `ip-dst` (the IP destination address) is revealed, as shown below:

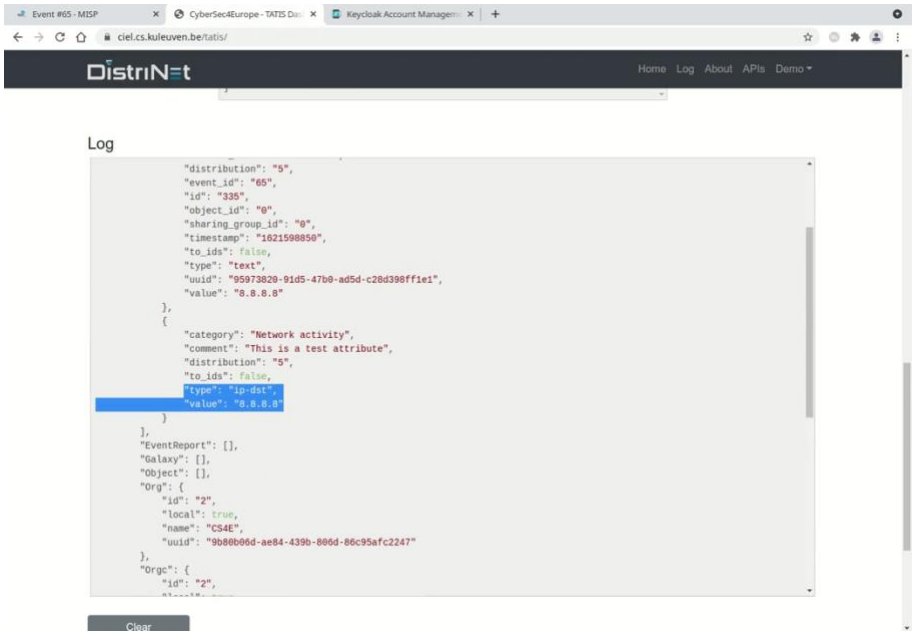

Figure 11 Decrypting the CP-ABE protected MISP attribute through the TATIS dashboard.

Programmatically interacting with TATIS through PyMISP

The video demonstration illustrates the same sharing process for a more sophisticated scenario involving network threat detections. In that example, the PCAP files and the analysis of external IDS tools are shared in a confidential and privacy-preserving manner through TATIS within MISP. Furthermore, there

is no need to use the TATIS dashboard. TATIS offers the necessary functionality to reuse existing PyMISP code samples (PyMISP[8] is a library to directly interact with MISP) with little modification.

```
from pymisp import PyMISP, MISPEvent, MISPAttribute, MISPObject
import requests
import json
from pathlib import Path

import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

if __name__ == '__main__':
    base_url = 'https://ciel.cs.kuleuven.be'
    auth_url = base_url + '/auth/realms/cybersec4europe/protocol/openid-connect/token'
    tatis_url = base_url + '/tatis/api/'
    misp_url = tatis_url + 'misp/'

    # Get an OAuth 2.0 access token from Keycloak end-point to communicate with TATIS reverse
    # proxy APIs.
    # NOTE: For the sake of convenience, we enabled the OAuth password grant type. For security
    # purposes, one should use the OAuth authorization code grant instead!!!
    username = "**********"
    password = "**********"

    data = {
        "username": username,
        "password": password,
        "client_id": "app-authz-misp-public",
        "grant_type": "password"
    }

    response = requests.post(auth_url,
                             headers={
                                 'Content-Type': 'application/x-www-form-urlencoded'
                             },
                             data=data)
    jwt = response.json()

    access_token = jwt["access_token"]

    misp_key = 'Bearer ' + access_token
    misp_verifycert = False

    misp = PyMISP(misp_url, misp_key, ssl=misp_verifycert, debug=False)

    ##############################################################################################

    # Create a new MISP event
    event = MISPEvent()
    event.distribution = "0"
    event.threat_level_id = "1"
    event.analysis = "0"
    event.published = False
    event.info = "This is a PyMISP test event via TATIS"
    event = misp.add_event(event, pythonify=True)

    # Fetch the same event again
    event = misp.get_event(event.id, pythonify=True)

    # Create a new MISP attribute
    attr = MISPAttribute()
    attr.category = "Network activity"
    attr.type = "ip-dst"
    attr.distribution = "5"
    attr.comment = "This is a PyMISP test attribute via TATIS"
    attr.value = "8.8.8.8"
    attr.to_ids = False
    # Add the attribute to the MISP event
    misp.add_attribute(event, attr, pythonify=True)

    # Create a new MISP object
    obj = MISPObject(name='security_event_object', strict=False)
    obj.template_uuid = "d2f7910b-f757-4370-9db1-cfa3e89c20b8"
    obj.template_version = "1"
    obj.sharing_group_id = "0"
    obj.description = "CS4E Security Event Object"
```

[8] https://www.circl.lu/doc/misp/pymisp/

```
obj.distribution = "5"
obj["meta-category"] = "network"

# Add attributes to MISP object
obj.add_attribute("ip_src", value="8.8.8.8", type="ip-src")
obj.add_attribute("ip_dst", value="192.168.0.1", type="ip-dst")
obj.add_attribute("ip_dst_port", value="8888", type="port")
obj.add_attribute("anomaly_score", value="0.99", type="text")
obj.add_attribute("creation-date", value="2021-05-19T12:20:00.123456+0000", type="datetime")
obj.add_attribute("attack_type", value="{\"KULIDS\": {}}", type="text")
obj.add_attribute("pcap_file", value='anomaly.pcap', type="attachment",
                  data=Path("898457.pcap"), expand='binary')
obj.add_attribute("anomaly_details", value='anomaly.json', type="attachment",
                  data=Path("anomaly.json"), expand='binary')
obj.add_attribute("verified", value="0", type="boolean")

# Add object to MISP event
misp.add_object(event, obj, pythonify=True)

# Fetch event as JSON object, not as Python object
event = misp.get_event(event.id)
# This will print event with obfuscated attribute
print('*** PRIVATIZED EVENT ***')
print(json.dumps(event, indent=4))
print()
print()

##########################################################################################

# Retrieve CP-ABE private key to decrypt privatized MISP attributes
response = requests.get(tatis_url + 'privkey',
                        headers={
                            'Authorization': misp_key,
                            'Content-Type': 'application/json'
                        })
privkey = response.json()

##########################################################################################

# Request TATIS to retrieve the same event while decrypting it using the CP-ABE decryption key
# NOTE: You should only share your CP-ABE private key to trusted clients. For the sake of
# simplicity, we assume that every user/organization hosts its own local TATIS reverse proxy to
# simplify the integration with PyMISP and the identity mapping between MISP and TATIS:
#   - each user/organization would have its own MISP authkey that TATIS uses as the reverse proxy
#     to MISP
#   - each user/organization has its privacy policy enforced by TATIS

data = {
    "key": privkey["bytes"]
}

response = requests.post(misp_url + 'decrypt_events/' + event['Event']['id'],
                         headers={
                             'Authorization': misp_key,
                             'Content-Type': 'application/json'
                         },
                         json=data)

decrypted_event = response.json()
print('*** DECRYPTED EVENT ***')
print(json.dumps(decrypted_event, indent=4))
```

Figure 12: Code example illustrating how to interact with TATIS through PyMISP.

For more details and a demonstration of TATIS with the network security example, we refer to the online video listed in section 3.1.1.

Anonymizing objects from MISP events with PP-CTI Sharing

TATIS can use the Privacy-Preserving CTI Sharing service as an external module to apply PETs to the information before sharing. The PP-CTI asset provides a REST API for TATIS to interact with. The input of the service is structured as a JSON object. It accepts events with objects like the one depicted in Figure 3 and policies in the format presented in Figure 13, which are contained in the JSON privacy policy managed by TATIS.

```
{
    "Event": {},
    "Privacy-policy":{
        "attributes": [],
        "creator": "alba.hita@um.es",
        "organization": "UMU",
        "templates": [
            {
                "attributes": [
                    {
                        "name": "pcap_file",
                        "type": "IDENTIFYING"
                    },
                    {
                        "name": "ip_src",
                        "type": "INSENSITIVE"
                    },
                    {
                        "name": "ip-dst",
                        "type": "QUASI_IDENTIFYING"
                    },
                    {
                        "name": "ip-dst-port",
                        "type": "QUASI_IDENTIFYING"
                    },
                    {
                        "name":"security_event_object",
                        "pets":[
                            {
                                "Scheme": "k-anonymity",
                                "metadata": {
                                    "K": 2
                                }
                            }
                        ],
                        "uuid": "d2f7910b-f575-4370-9db1-cfa3e89c20b8"
                    }

                ],
                "version": "1.0.0"
            }


        ]
    }
}
```


Figure 13: Privacy Policy for PP-CTI Sharing asset

After receiving the threat event and privacy policy from TATIS, PP-CTI performs the PET mechanisms to obfuscate the information. With the policy shown in Figure 13, the PET applied is k-anonymity.

K-anonymity (Samarati 2001) is one of the most extended and used privacy-enhanced mechanisms. It aims to obfuscate the identity attributes of a data record for a set of k-1 records, making this information unlinkable but still useful. In this example (Figure 13), identity attributes are related to the destination addresses (`ip-dst` and `ip-dst-port`), which are meant to be protected.

To apply this mechanism, it is necessary to create a hierarchy level of obfuscation to apply more restricted anonymization or a lesser one. In this case proposed, the attributes `ip-dst` and `ip-dst-port` will be anonymized with the following hierarchies.

```
ip-dst: (10.20.30.40, 10.20.30.0, 10.0.0.0, 0.0.0.0)
ip-dst-port: ("<1000,>1000";"<500,<1000,>1000")
```

Depending on the number of attributes `ip-dst` and `ip-dst-port` are in the threat event, the PET will use a hierarchy more restrictive or less. The new event will contain the same attributes but with the information from these attributes obfuscated.

3.2 Scenario 2. Enrichment of CTI

3.2.1 Summary

In this section we demonstrate the cooperation among TDSs and honeypots. The main idea consists in enriching the information on detected threats with further details provided by different TDSs. Moreover, the honeynet allows for gathering further attack instances which will be used in the learning phase of the ML-Based TDSs to improve their effectiveness.

A typical flow within the proposed scenario includes the following steps: (1) An IDS documents an attack and updates its corresponding MISP with threat intelligence (events and attributes) describing the attack. (2) The information concerning the attack is shared between entities, possibly after enhancing its confidentiality and privacy, and an operator, using TIP information, deploys a honeypot configured to enrich information concerning the new attack. (3) further information concerning the uploaded events or external events potentially relevant for the monitored network infrastructure can be collected by exploiting the data enrichment platforms. In addition, when the information on a new intrusion is gathered by the honeynet, once again these data are shared with the MISP via a custom MISP object.

3.2.2 Assets Roles

3.2.2.1 EBIDS

EBIDS acts both as a data provider and consumer. As a consumer, it can retrieve custom security events (either produced by other TDS or logged by custom honeypots) from the MISP and exploit them to improve the detection capabilities of the underlying detection model. This is achieved by feeding it with new labeled data concerning the attacks represented by the security events. As a producer, EBIDS can identify new attack instances from which to build and deliver the related security events. Such events can hence be consumed by other TDS or exploited to deploy specific honeypots tailored for the underlying attacks. Furthermore, it can analyze security events already stored within the MISP and enrich them with additional attack_type labeling.

In Figure 14, we describe the interactions among the main components. Basically, we developed three macro-modules to provide import/export facilities: (i) Log Extraction and Transformation (Honeypot Log ETL) module, (ii) Threat Importer and (iii) IDS Event exporter. The first one is devoted to gathering data (e.g. log files) from the honeypot pool and providing them to the MISP, via Rest Server. The Threat Importer collects data (pcap files) from the security events and prepares them for the learning phase of EBIDS. Finally, the IDS Event Exporter produces security events discovered by EBIDS and shares them with the MISP. All communications within the platform occur via Rest Server.

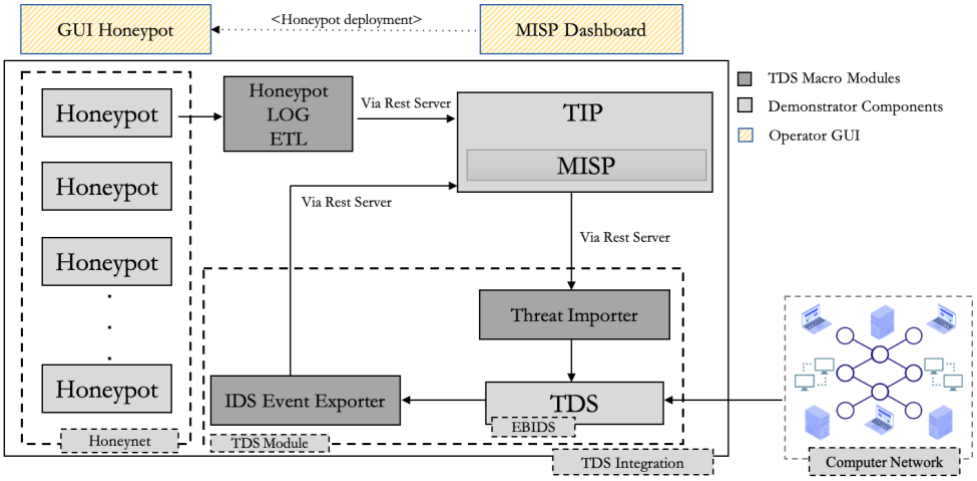

Figure 14 EBIDS Role and integration

EBIDS works in a continuous stream that includes two phases: learning and deployment. In the former, we train the classification model which is then exploited in the latter to detect anomalous behaviours and store them under the form of security events. In the following, we provide a detailed description of these phases highlighting the interfaces and the components required for the cooperation of MISP, Honeypots and IDSs.

Learning phase. Figure 15 shows the logical flow of the learning phase. In this phase, a set of classifiers are trained against Threat and Regular Traffic flows stored in a local database (named PCAP DB in figure). The software component, named DB Manager, is devoted to filling the PCAP DB with relevant threat events extracted from the TIP and normal traffic flow examples gathered from the computer network. Moreover, it updates the DB by sampling and discarding out-of-date "normal traffic" and possibly also obsolete "threat" data. The examples stored in the PCAP DB are first processed through a data extraction tool (*CICFlowMeter* (Lashkari, et al. 2017) in the figure) that extracts statistical summaries from the pcap network information. The resulting relational representation then feeds a training set, which is exploited to update EBIDS. The Learning Phase is periodically performed to keep the model up-to-date.

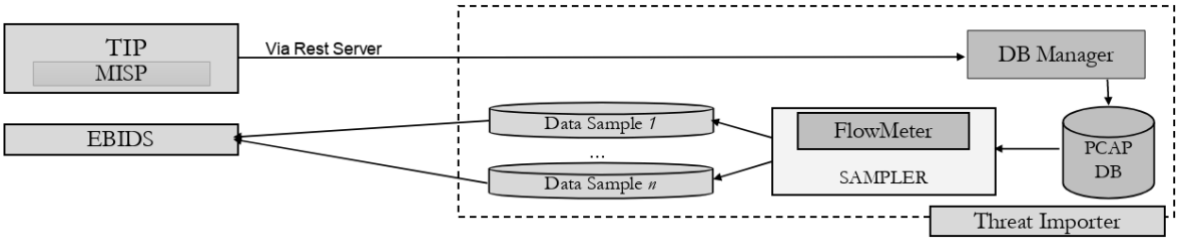

Figure 15 Threat Importer components

Deployment and usage. The deployment phase exploits the learned model to detect anomalous behaviours and store them as MISP security events. Basically, our scenario refers to the classical Network IDS (NIDS) application (see Figure 16): NID analyses the network traffic extracted from a network exploiting specific tools (e.g. *packet capture*) with the aim to monitor the devices connected to a network.

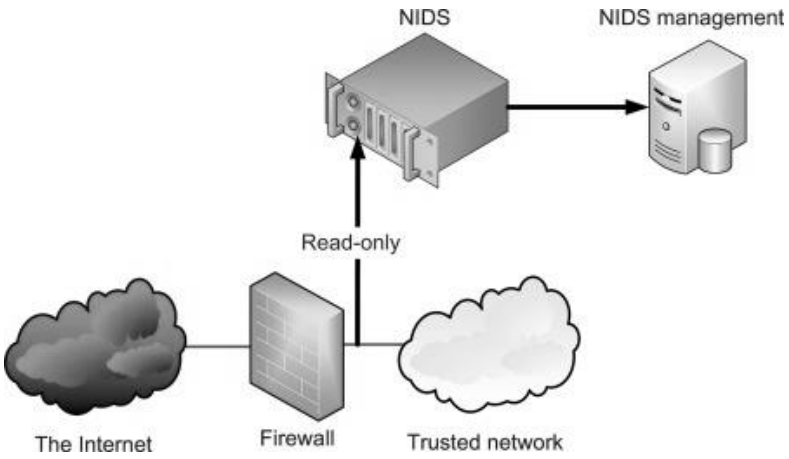

Figure 16 Network IDS Deployment Diagram

Figure 17 illustrates the components of the underlying architecture and the typical process: the network flow is stored into a PCAP file and its preprocessing is provided to EBIDS for the threat detection. If an anomaly (i.e. a possible new threat) is detected, then it is shared with the TIP via IDS Event Exporter. Both the pcap and the preprocessed data are stored within the event, through the `pcap_file` and `anomaly_details` attributes. DB Manager also manages the update procedure of the normal traffic, stored into the PCAP DB for the learning of EBIDS. Upon request, pcap files imported from MISP security events can be analyzed and an updated MISP Security Event object is created and stored into the TIP with the additional labeling provided by EBIDS. This way, false positive rate can be reduced by means of collaborative analysis.

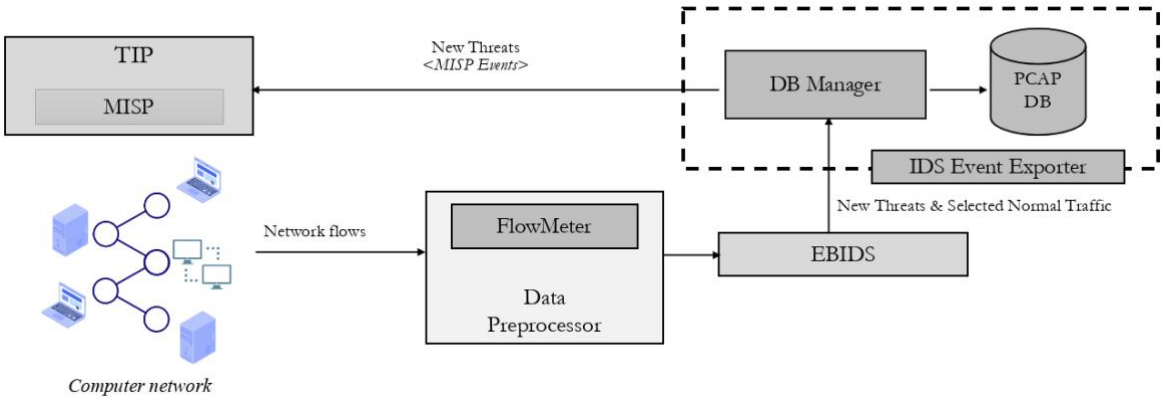

Figure 17 IDS Event Exporter Components

3.2.2.2 NetGen

NetGen is used in this scenario to classify the suspicious traffic detected by anomaly detection tools such as EBIDS. NetGen consists of two main components: an IDS generator and the set of trained IDSes. In this scenario, a collection of pre-trained IDSes will be used to detect malicious traffic and activities on the network.

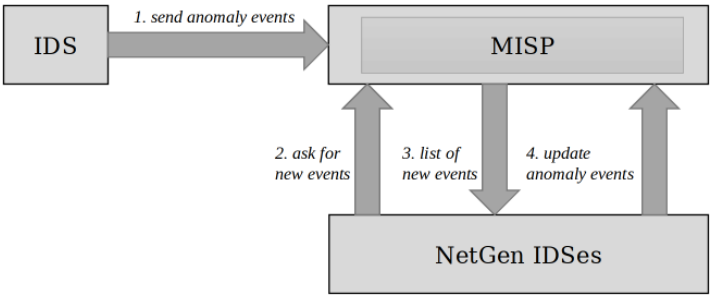

Figure 18 Interactions between NetGen and the other scenario components

As shown in Figure 18, the IDSes trained with the NetGen framework will periodically (e.g., once a day) query the MISP nodes (part of the TIP component) to request all the most recent MISP security events detected by any anomaly detection tool (e.g., EBIDS). NetGen will then download the events and their attached PCAP files and it will analyse them to better identify the attack types behind the suspicious activity (e.g., DDoS attack, SQL injection, web scraping attempt). Successively, NetGen will update the MISP security event by adding the classification results to its data and it will push the updated MISP object to the TIP. For instance, the following JSON code is an example containing an anomaly detected by EBIDS and classified as a Slowloris DoS attack by NetGen:

```
{"EBIDS": {
      "version": "0.2",
      "reference": "https://github.com/massimo-guarascio/cs4e_ebids_asset",
      "attacks": [{
          "attack_type": "ANOMALY",
          "confidence": "0.9933746"
        }]},
 "NetGen": {
      "version": "0.1",
      "reference": "https://github.com/daniele-canavese/netgen",
      "attacks": [{
          "attack_type": "dos_slowloris",
          "confidence": "0.91461"
        }]
  }
}
```

Figure 19 NetGen updates SEO attachment

NetGen, similarly to EBIDS, will add its classification results and its confidence level (a number between 0 and 1 stating the classification posterior probability) to the MISP security event.

3.2.2.3 TIE

The role of the Threat Intelligence IntEgrator (TIE) in this scenario is to analyze any security event received through the MISP instance and enrich it with a context awareness score value that represents its relevance and actionability in the specific infrastructure of the organization. This is mainly relevant in case of security events shared among different organizations or indicators of compromise received from external OSINT data sources.

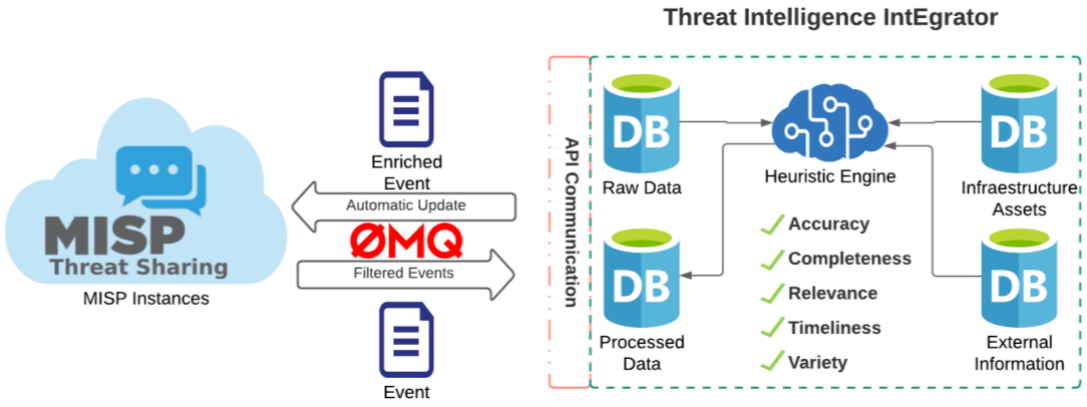

Figure 20 TIE Components Overview

The Threat Intelligence IntEgrator (TIE), as shown in Figure 20, is directly connected to the MISP instances through a ZeroMQ channel. Through this communication, TIE is able to obtain all the new events that are generated and those previous ones that are published in the MISP ZMQ queue in order to process them. The retrieval is done by filtering these new events through the use of specific tags. Thus, once the event arrives at the TIE, it is processed with the heuristic engine of the system. The heuristic engine uses different sources to calculate the score, among which are the infrastructure's asset database and data from external sources (Twitter, blogs, etc.). The heuristic engine evaluates and gives weights to the five characteristics of each event (Accuracy, Completeness, Relevance, Timeliness, Variety). With these, it calculates the final score that is added to the initial event. Finally, the event with the score is stored in a database of processed information and then sent to the MISP instance to be updated. Therefore, we foresee that the Threat Intelligence IntEgrator (TIE), will be very useful within the scenario, taking the role of an evaluator of external threats that may affect not only the network but also systems and services in general within the network. Through the new attribute, added by the TIE within MISP events, the security analyst can take measures to reduce possible attacks from threats that may come from outside the monitored network. The following JSON (Figure 21) shows the structure of the TIE's score attribute added within a MISP event.


```
{
  "description": "CVE-2019-3923 (nessus): Nessus versions 8.2.1 and earlier
    were found to contain a stored XSS vulnerability https://t.co/yht0hgNYF4",
  "threat_score": 3.643410852713178,
  "cve_id": "CVE-2019-3923",
  "node": "hash4",
  "application": "nessus2"
}
```


Figure 21 TIE's IoC JSON structure

3.2.2.4 INTELFRAME

IntelFrame aims to work with the intrusion detection components by enhancing / maintaining the detection performance of a detector / classifier, through helping choose a suitable learning algorithm or parameter settings within a time period. As shown in Figure 22, IntelFrame can collect required data from the IDS pool, e.g., labelled events, alarms, and prepare its own dataset, which can be used for tuning the parameters for deep learning (DL) algorithms. Then, an optimized DL algorithm will be returned to the IDS pool to enhance the detection performance.

The framework can overall provide several advantages:

- The algorithm can be selected intelligently in a period of time.

- The algorithm with better performance will be selected.

- Algorithms can also be used to decide an anomaly via data fusion & aggregation, i.e., considering the outputs from all used algorithms.

Figure 22 shows how to explore a suitable setting for deep learning. As threat events could be shared between honeypots and IDS detectors, the network status may be changed after a time period, which needs an update of the pre-established model. How to set up the parameter settings is important to maintain the effectiveness of detection.

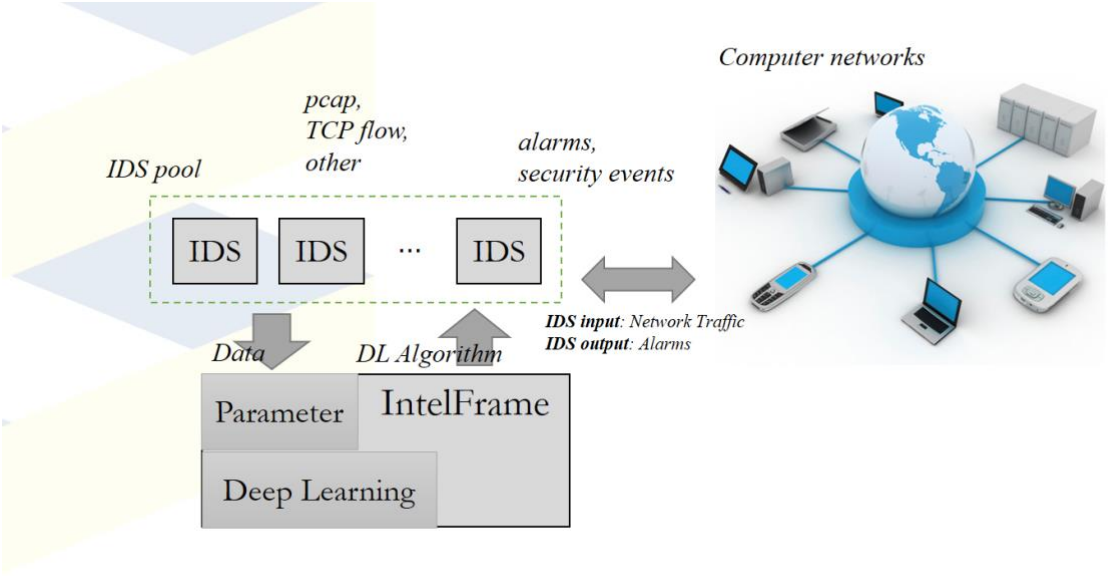

Figure 22: The interaction between IntelFrame and IDS pool.

3.2.2.5 HADES

The role of HADES as a data enrichment module is to provide additional information about potential malware files detected during the operation of the platform. Such information is provided by HADES internal components and modules (Cuckoo Sandbox, Suricata, Snort), and include data such as i) signatures related to the behaviour of the file, such as APIs called and changes in the registry information, ii) network domains and URIs that are accessed by the file, and iii) Snort and Suricata alerts generated during the execution of the file.

Figure 23: HADES components overview

The integration of HADES within the architecture is shown in Figure 23. This asset retrieves MISP events that contain information about files (including the files themselves) from MISP instances, either using a “pull” approach (i.e. use a publish/subscribe channel to obtaining new events) or a “push” approach (i.e. periodically querying the MISP instances to retrieve the latest events). The files will then be loaded and analyzed within HADES. Finally, all the information gathered from HADES components will be used to enrich the event, which will be then sent back to the MISP instance. Note, however, that HADES can take a relatively long time (e.g. 30 minutes) to analyze large malware samples. It is then necessary to consider this delay when configuring this asset.

3.2.2.6 Reliable CTI

The Reliable CTI Sharing asset has the role of enriching the information about the trustworthiness of the source of it. It acts as an external service to the TIPs.

The asset receives threat events from these platforms, which are JSON-formatted, through the use of publisher/subscriber communication channels, such as ZeroMQ or through an API REST. It inspects all the data included in these events, with careful attention to how they have been sent, who has sent them and created them, the trust in the intrinsic relationships of the organisations, and what elements of privacy and security they incorporate. Depending on those variables, including the behaviour of who has sent and created them, it scores a multi-dimensional value. This value informs about the trust the information has regarding our organisation. It stores the global trust in a database for better performance, ready to query from any other service or to be scored by any TIP or external proxy (such as TATIS).

For interaction with MISP, a module is being developed, as shown in Figure 24. It interacts with the Reliable CTI asset and creates a new object with the calculated trust score. This new object is intended to enrich the threat event by adding the trustworthiness evaluation. The trust objects in the events could be later used to know the score and use the threat event knowing this information.

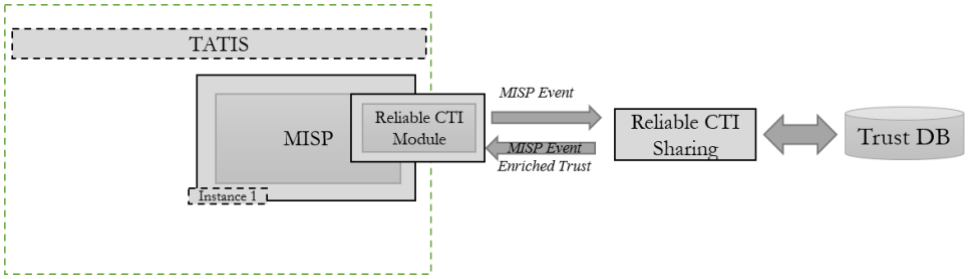

Figure 24: Reliable CTI Sharing components

3.2.3 Demonstrations

3.2.3.1 Adaptive collaborative information sharing

The idea behind this demonstration is shown in Figure 25. The starting point is represented by the monitored system. Here, traffic flows from the computer network are periodically analysed by the components of the TDS Layer. In this scenario, a specifically anomaly detector (TDS_1 in figure), analyses the pcap files including a specific logged traffic flow and detects an anomaly. A MISP security event is then produced and stored on MISP. The event is then distributed in the TIP, a different IDS (TDS_2 in figure) accesses the event, analyses the embedded pcap files and provides an additional labelling (in agreement with the labelling from TDS_1) within the *attack_type* field. On the other hand, the TIE component can verify and qualify if any of the events that reach MISP through external sources or shared by IDS deployed in other entities match the entity's internal architecture components. This verification and scoring of events allow the analyst to receive more information about possible security risks present in her network. The TIE score to the analyzed events is added to the original event, updating it. The updated MISP objects, now with two consensus labels and the TIE's threat score, is analysed by an expert who validates the threat classification. The validated event can now be retrieved from other IDS (e.g., TDS_n in the figure) and included in their training set for improving the robustness of the underlying classification model. Regarding the last phase of this use case, addition of further information to the threat events and complementary to the TIE asset, Reliable CTI asset enriches the information with a trust score. The calculation of trust is unique to an organisation, as it is based on the relationship and interaction of the organisation with the one that created the event and who sent it. The trust score of

this event is stored within the event, so the analyst can use it as a bias and prioritisation of the information it contains.

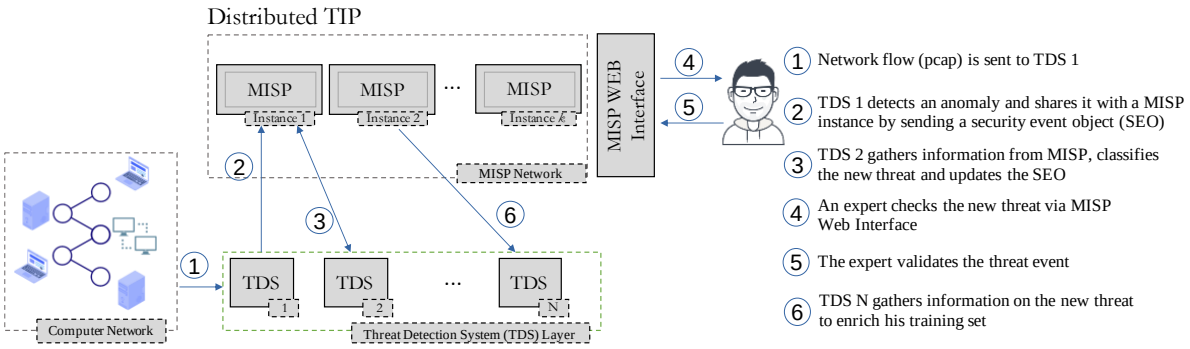

Figure 25 An example of execution flow.

To demonstrate the capabilities of our architecture, we performed multiple attacks on a virtualized environment, shown in Figure 26.

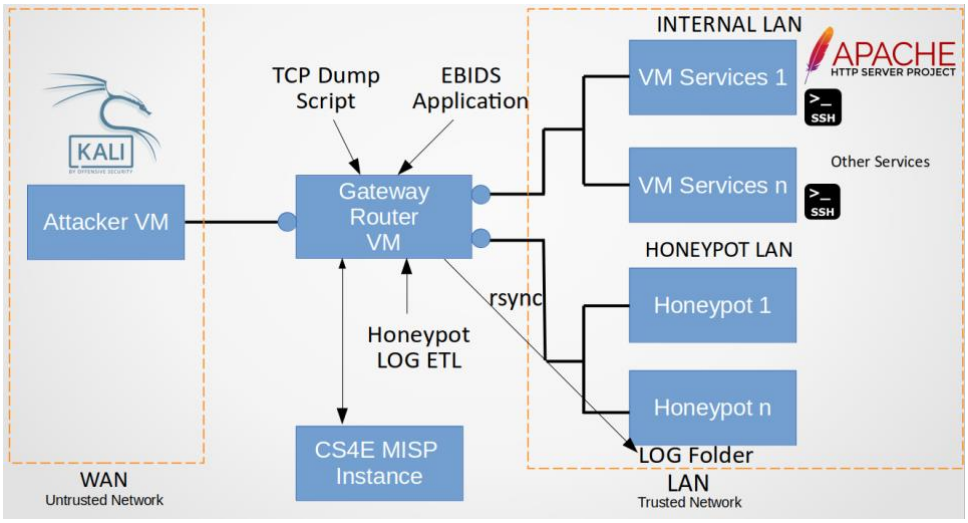

Figure 26 Testbed Environment

Basically, the proposed testbed includes two sub-nets interfacing through a Gateway Router. In more detail, it is composed of an untrusted network representing the source of the attacks and a monitored trusted network, target of the attackers. The Trusted Network is segmented in 2 parts: the “internal LAN” containing machines that host external and internal service plus classic office workstations and the “honeypot LAN” containing only honeypot machines.

The gateway router includes an internal IDS (like Snort or Suricata) that evaluates every request. If the request is suspicious it is routed to honeypot LAN, otherwise the request is forwarded to the internal LAN.

The main components of the testbed are listed as follows:

- **Attacker VM**. This VM runs a Kali Linux distribution and it is used to perform different types of attack to the trusted network.
- **Gateway Router VM**. An Ubuntu 20.10 VM simulating the network router and firewall behaviour. The whole EBIDS suite (application, learning and threat importer), the Flowmeter application (required to extract network traffic statistics) and Honeypot LOG ETL (devoted to exporting on MISP the Honeypot detected threat) are deployed on this VM.

- **CS4E MISP Instance**. An Ubuntu 18.04 virtual machine running an instance of MISP platform.
- **Services VMs**. A pool of VMs hosting different type of services target of the attackers. In our test case we deployed two virtual machines (based on Ubuntu 20.10) with Apache2 server as available service.

- **Honeypot VMs**. A pool of VMs hosting various Honeypots services. Every VM can host multiple Honeypots and in this demonstrator, we used a solution based on T-Pot Honeypot distribution (https://github.com/telekom-security/tpotce). T-Pot runs on Debian (Stable) and includes several dockerized versions of different honeypots as well as multiple tools to discover, monitor and analyze intrusion attempts to dockerized honeypots.

The EBIDS classifier is trained against four datasets extracted from CICIDS 2017, a well-known IDS benchmark dataset containing both benign and attack instances belonging to different classes. Moreover, the environment described in Figure 26 has been exploited to perform further attacks and enrich the starting training set. The classification model is incrementally learnt against different data chunks: multiple classifiers are trained on data gathered in different time windows. Out of date benign data are periodically discarded while the attacks are kept in the knowledge base as positive examples. Network traffic flows are gathered by the Gateway and stored in pcap format.

Relevant statistical features summarizing the network behaviour are extracted by CICFlowMeter and stored in CVS format to be provided as input for the learning phase (see Figure 27). Specifically, 80 network traffic features are automatically extracted including duration, number of packets, number of bytes, length of packets, etc. Figure 28 shows a view of the extracted data.

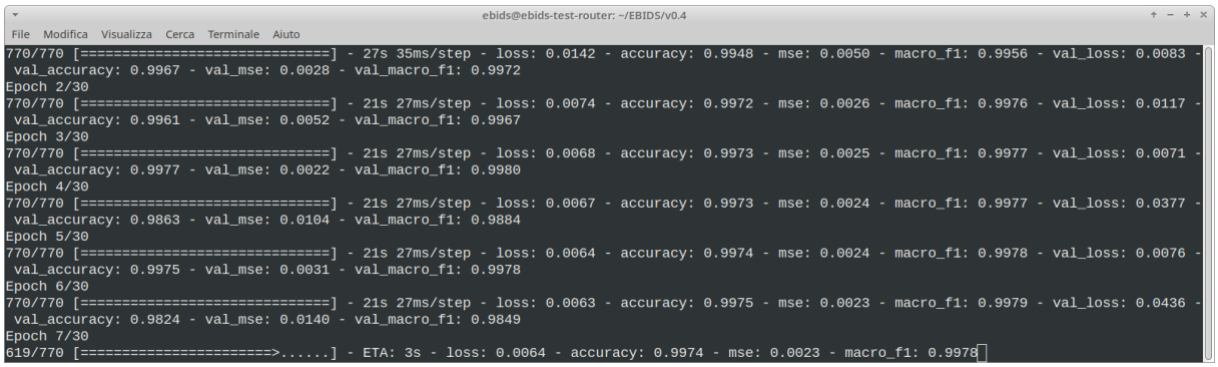

Figure 27 EBIDS learning phase

Destination_Port	Protocol	Timestamp	Flow_Duration	Total_Fwd_Packets	Total_Backward_Packets	Total_Length_of_Fwd_Packets	Total_Length_of_Bwd_Packets	Fwd_Packet_Length_Max	Fwd_Packet_Length_Min	Fwd_Packet_Length_Mean
80	6	13/04/2021 09:32:11 AM	15400	7	5	326	11173	326	0	46.57142857142857
80	6	13/04/2021 09:32:13 AM	2304152	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:18 AM	2009505	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:22 AM	2058696	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:26 AM	2008477	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:30 AM	2007972	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:11 AM	21433032	5	6	381	482	324	0	76.2
80	6	13/04/2021 09:32:11 AM	21439185	5	6	372	482	324	0	74.4
80	6	13/04/2021 09:32:11 AM	21424480	5	6	366	482	324	0	73.2
80	6	13/04/2021 09:32:11 AM	21411378	5	6	385	482	324	0	77
80	6	13/04/2021 09:32:11 AM	21405794	5	6	364	482	324	0	72.8
80	6	13/04/2021 09:32:11 AM	21420379	5	6	409	482	324	0	81.8
80	6	13/04/2021 09:32:33 AM	2964	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	3186	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	3166	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	2268	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	3337	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	2551	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:11 AM	21740429	5	6	373	482	324	0	74.6
80	6	13/04/2021 09:32:33 AM	290	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:32 AM	1478232	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:11 AM	22792616	5	6	400	482	324	0	80
80	6	13/04/2021 09:32:34 AM	245	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:11 AM	23802995	5	6	403	482	324	0	80.6
80	6	13/04/2021 09:32:35 AM	260	2	0	0	0	0	0	0
80	6	13/04/2021 09:32:33 AM	1999507	4	2	326	0	326	0	81.5
80	6	13/04/2021 09:32:11 AM	24736393	5	6	372	482	324	0	74.4

Figure 28 CICFlowMeter Features

To evaluate the performances of our classifier in recognizing anomalous behaviours occurring on the networks we performed different attack types. The ensemble model is composed of K classifiers (one for each time window), then some attacks are performed in subsequent time windows. The application module, based on the EBIDS model, analyses each CVS generated by CICFlowMeter and if an

anomalous instance is detected then it is shared with the MISP by generating a new Security Event object. In Figure 29, we show the output of the detection: two threats are discovered and sent to the MISP server.

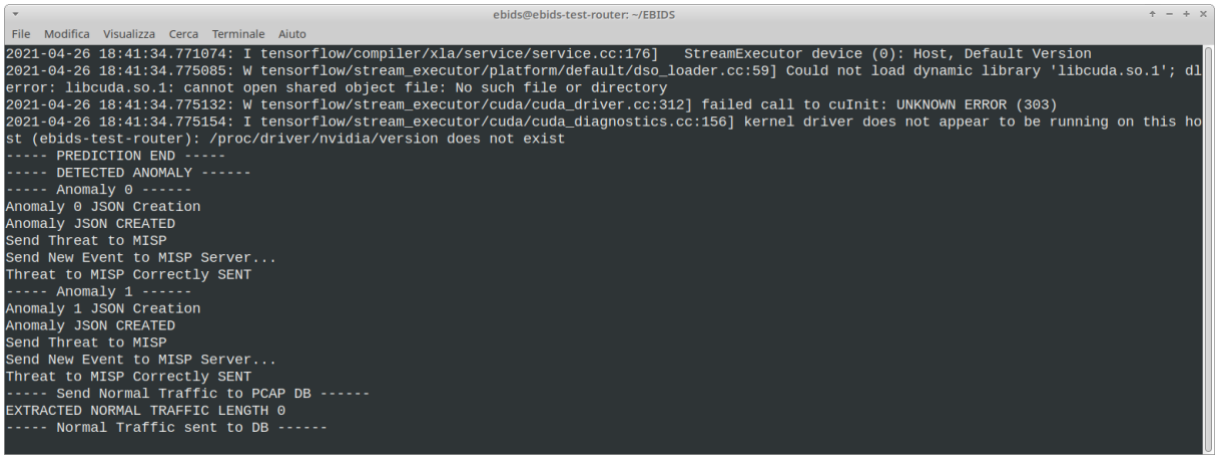

Figure 29 Classification Application Output

Figure 30 shows the details of a threat on the MISP server.

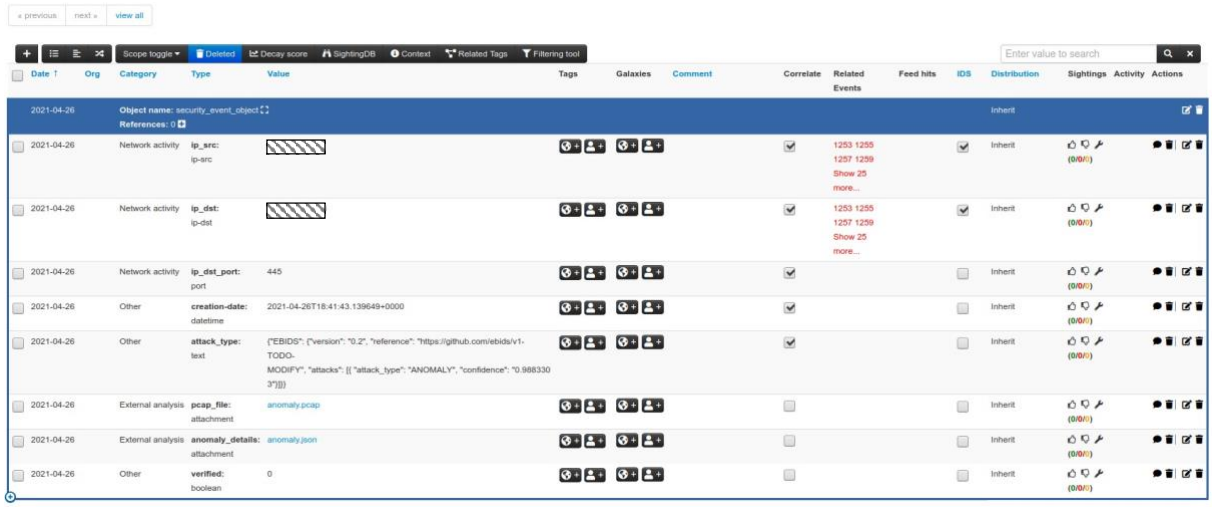

Figure 30 Example of MISP Server Threat

Attack Detection

Slowloris attack. The idea behind the Slowloris attack consists in sending legitimate HTTP requests and reading responses slowly aiming to keep as many connections as possible in the active state. The attack has been simulated by using the SlowHttpTest application with the following parameters:

- *target number of connections*: 600
- *interval between followup data in seconds*: 10
- *connections per seconds*: 200
- *timeout to wait for HTTP response on probe connection*: 2
- *max length of each randomized name/value pair of followup data per tick*: 24

The model is able to recognize a high percentage of anomalous flows yielded by the attack (97% of recall) simultaneously arising a limited number of false positives (1% in terms of false positive rate). The identified flows are then stored on MISP (Figure 31) to be classified and verified.

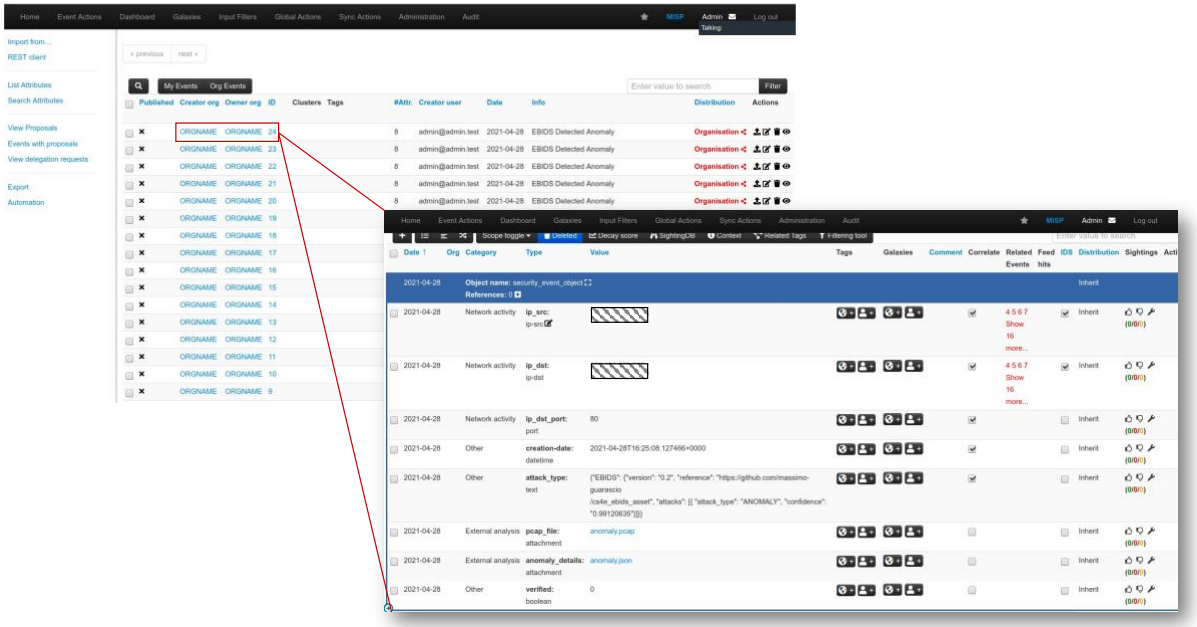

Figure 31 Slowloris anomalous flows stored on MISP

GoldenEye attack. GoldenEye is a HTTP DoS Test application exploiting 'HTTP Keep Alive + NoCache' as attack vector. This tool can open several parallel connections against a URL to check if the web server can be compromised. The attack tries to open multiple connections and so to saturate the server capacity generating a denial of service. Also in this case, the model exhibits good performances in terms of predictive accuracy with high recall and low false alarm rate. The identified flows are then stored on MISP (Figure 32) to be classified and verified.

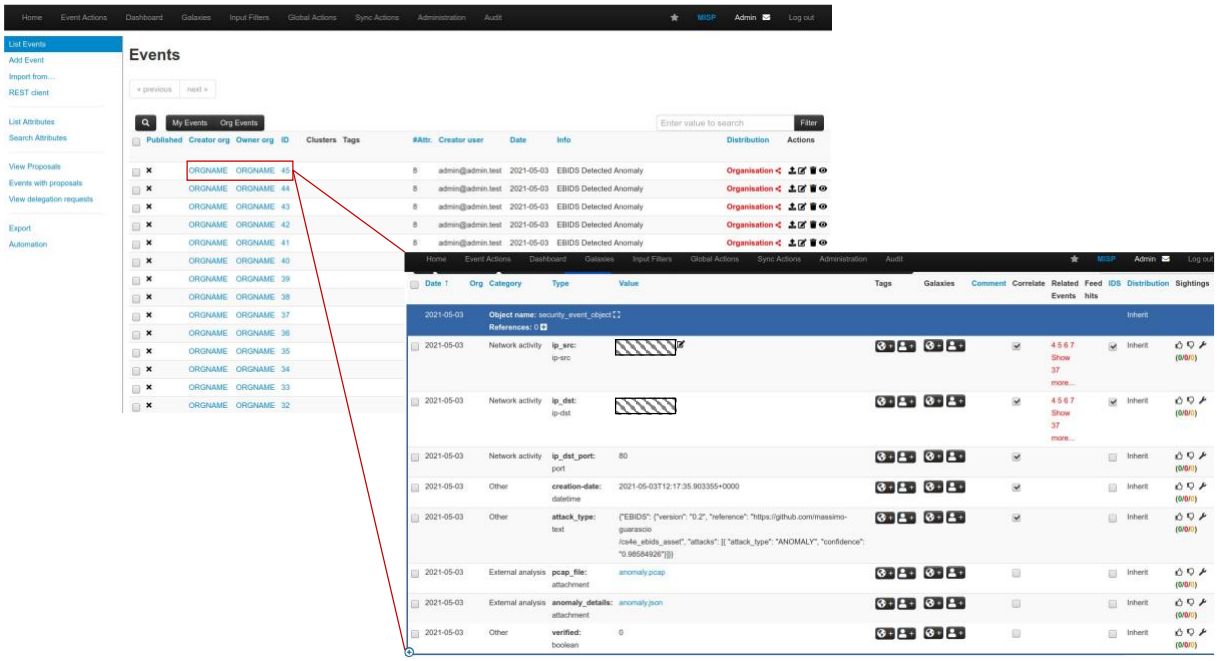

Figure 32 Goldeneye anomalous flows are stored on MISP

Attack Classification

NetGen will be used to enrich the anomalous MISP events detected by EBIDS and stored on MISP by adding proper classification and confidence levels. In the adaptive honeypot NetGen will focus on detecting two types of denial-of-service attacks (Slowloris and Goldeneye), though it is powerful and flexible enough to be used to classify a much greater variety of attacks and suspicious activities. Figure 33 shows a browser window depicting an updated event (related to a GoldenEye attack). This operation can be easily performed periodically (e.g., every day) by scheduling a proper cronjob.

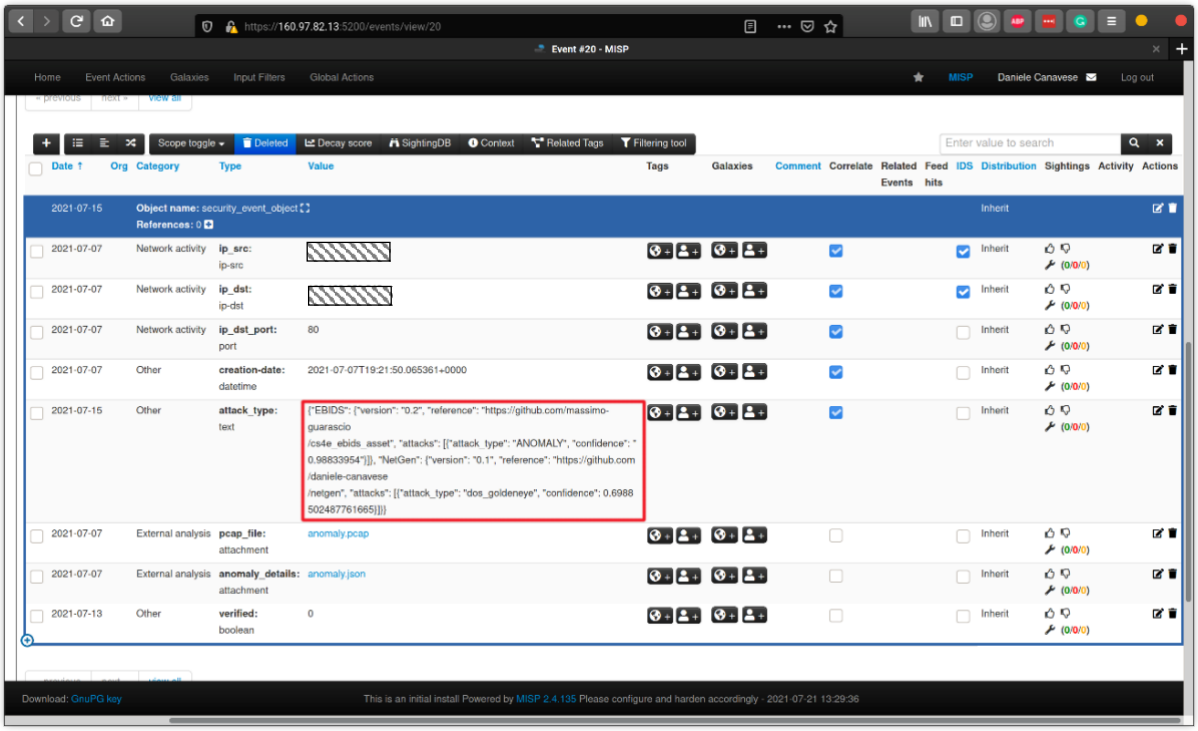

Figure 33: Example of anomalous MISP security events with the NetGen classification

Training and initial testing. NetGen will be trained and tested using a mix of real traffic data (PCAP files) coming from the CIC IDS 2017 dataset. NetGen was used to train an IDS able to distinguish between benign traffic, Slowloris and GoldenEye denial-of-service attacks, thus discriminating also between the tools used to perform the attack. Note that the "benign traffic" label is theoretically unnecessary if the user assumes that an anomaly is always an attack, but its presence can be viewed as an auxiliary consistency check for whole attack/anomaly detection workflow. NetGen internally makes use of the tstat network statistics and analysis tool to transform a TCP connection in a sequence of statistics that are then used to train and optimize a variety of machine-learning models. Table 2 shows some statistics on the training and test sets obtained by analysing the CIC IDS 2017 public data set.

metric	*training set*	*test set*
number of samples (TCP flows)	2546732	282971
mean confidence	0.973	0.917
accuracy	0.977	0.976
balanced accuracy	0.973	0.973
AUC	0.998	0.997
Matthews correlation coefficient	0.965	0.963

Table 2: NetGen IDS classification metrics

In the following paragraphs, the detection accuracy of some anomalies detected by EBIDS and stored on the MISP server are presented and discussed.

Slowloris attack detection. On a test set containing 3000 Slowloris flows detected by EBIDS and stored on the MISP server, NetGen was able to correctly classify 2951 of the connections, thus obtaining an accuracy of 98%. While, most of the mis-classified flows were categorized as GoldenEye attacks, a limited number of attacks (about 0.001%) were misclassified as benign traffic.

GoldenEye attack detection. Analogously, on a data set containing 3000 GoldenEye connections detected by EBIDS and stored on a MISP server, NetGen achieved a 97% accuracy, by correctly classifying 2923 flows. The remaining misclassified connections were mostly categorized as Slowloris flow and a limited number of attacks (about 0.001%) were classified as benign flows.

Event enrichment and information sharing

As stated in section 3.2.2.3, the TIE asset will constantly receive information from external and internal MISP instances. By receiving information from external instances, the TIE is able to quantitatively identify and rank threats from external sources and the risks of those threats to the internal network. Once the TIE has quantified the importance of external events to the internal network, it updates and publishes them for consumption by the other internal MISP instances. Thus, the publication of the updated event with the TIE score is propagated to the other instances and is consumed by other assets in the network to - if necessary - update defence rules that the network contains.

On the one hand, TIE contains an internal inventory database with information received about the monitored infrastructure in the organization (an extract is shown below in Figure 34). It allows the TIE to know about the components within the network that could be affected by any attack or recent discovery vulnerability. In particular, the information registered could be the name and version of the applications installed, servers' ip address, operating systems, network devices, IoT devices, etc.

```
{
  "nodes": {
    "hash1": {
      "name": "Web Server",
      "type": "Web Server",
      "object": "Web Server",
      "description": "Web Server",
      "software": [
        {
          "name": "Linux",
          "version": 5.1,
        },
        {
          "name": "Apache",
          "version": 2.2,
        }
      ],
      "alerts": [],
      "ips": [],
      "keywords": [
        "server",
        "web"
      ],
    },

    "hash2": {
      "name": "Desktop PC",
      "type": "endpoint",
      "object": "real",
      "description": "User desktop pc",
      "software": [
        {
          "name": "Windows 10",
          "version": 19042,
        },
        {
          "name": "Microsoft Office",
          "version": 17,
        }
      ],
      "alerts": [],
      "ips": [],
      "keywords": [
        "endpoint",
        "windows"
      ],
    }
  }
}
```


Figure 34 Inventory Database Example

On the other hand, MISP instance will be connected to external MISP instances from different organisations (National CERTs, Private Organisations, Public Organisations, etc.) in order to receive information about new vulnerabilities or security flaws that may affect the internal network (e.g., vulnerabilities related to services such as Apache HTTP). By receiving the external information, the TIE asset will automatically calculate the risk score of each received event related to any relevant asset within the network and known by the TIE's inventory.

The following screenshot, Figure 35, shows a received event related to a DDoS vulnerability. The discovered vulnerability is within the Linux Kernel; therefore, all the services inside the network that use Linux could be a potential attack target.

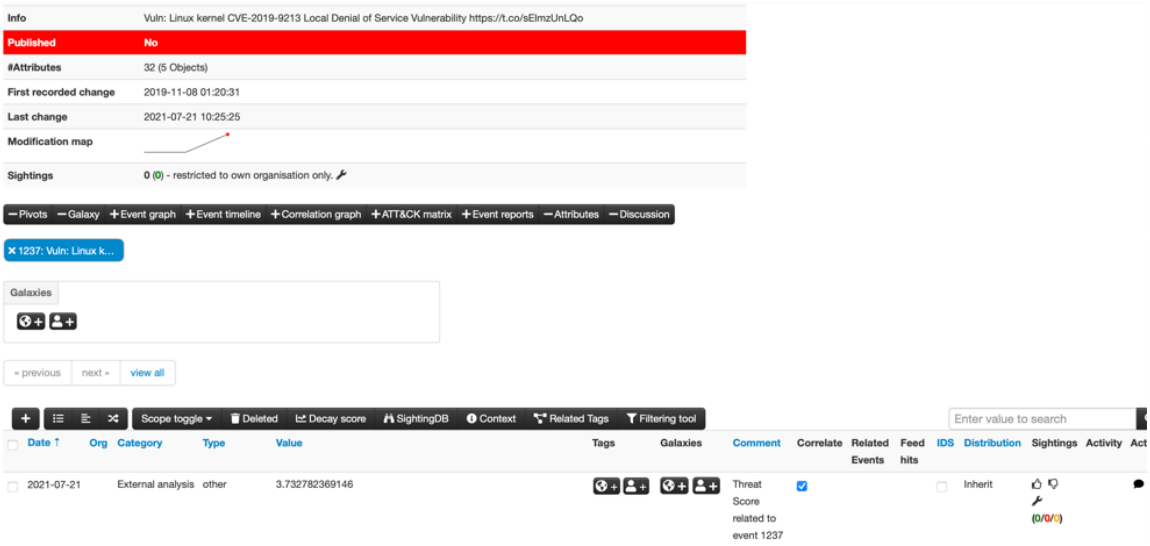

Figure 35 DDoS Vulnerability Event.

The TIE automatically calculates the threat score related to the received event and the previously known internal network. As is shown in Figure 36, the calculated score is added to the event. This will help the security analyst to address the vulnerability accordingly.

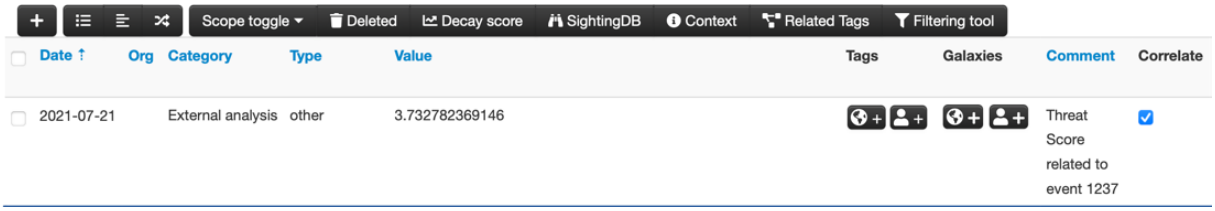

Figure 36: Event threat score.

The value of the score ranges between 0 and 5 and can be adapted with the configuration of the heuristics according to the needs of each infrastructure and system (Gonzalez-Granadillo, et al. 2019). This type of information enrichment allows analysts to give priority attention to the events that are most related to the infrastructure they manage.

3.2.3.2 Honeypot Threat Sharing

In addition to the information provided by the TDSs, the framework allows for gathering further attack examples by means of a honeynet so to enrich the knowledge base used for training of the ML-Based TDSs. In more details, each deployed honeypot gathers data concerning new intrusion in a log file (in json or xml format). Notably, the services provided by the honeypots seem legitimate, but they should be not used from anyone, therefore any attempt to access them can be considered as an intrusion and then labelled as attack. In the following a sample of the honeypot log file is shown.

```
{"eventid":"cowrie.session.connect","src_ip":"192.168.250.147","src_port":32962,"d
st_ip":"xxx.xxx.xxx.xxx","dst_port":22,"session":"bb8b736d0154","protocol":"ssh","
message":"New connection: 192.168.250.147:32962 (xxx.xxx.xxx.xxx:22) [session:
bb8b736d0154]","sensor":"032e50ff5c5a","timestamp":"2021-06-10T08:22:05.652729Z"}

…

{"eventid":"cowrie.login.failed","username":"root","password":"****","message":"lo
gin attempt [root/phil] failed","sensor":"032e50ff5c5a","timestamp":"2021-06-
```

```
10T08:22:15.307077Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}

…

{"eventid":"cowrie.command.failed","input":"ll","message":"Command not found:
ll","sensor":"032e50ff5c5a","timestamp":"2021-06-
10T08:22:25.771122Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}
{"eventid":"cowrie.command.input","input":"ls","message":"CMD:
ls","sensor":"032e50ff5c5a","timestamp":"2021-06-
10T08:22:27.434825Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}

{"eventid":"cowrie.command.input","input":"wget gmail.com","message":"CMD: wget
gmail.com","sensor":"032e50ff5c5a","timestamp":"2021-06-
10T08:22:34.778660Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}

{"eventid":"cowrie.session.file_download","url":"http://gmail.com","outfile":"dl/2
e5d16521a7625377d0b9ae6e488d7ccc1d55ef8f81b17e885264e83d9714cb8","shasum":"2e5d165
21a7625377d0b9ae6e488d7ccc1d55ef8f81b17e885264e83d9714cb8","message":"Downloaded
URL (http://gmail.com) with SHA-256
2e5d16521a7625377d0b9ae6e488d7ccc1d55ef8f81b17e885264e83d9714cb8 to
dl/2e5d16521a7625377d0b9ae6e488d7ccc1d55ef8f81b17e885264e83d9714cb8","sensor":"032
e50ff5c5a","timestamp":"2021-06-
10T08:22:35.990249Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}

…

{"eventid":"cowrie.session.closed","duration":39.99149441719055,"message":"Connect
ion lost after 39 seconds","sensor":"032e50ff5c5a","timestamp":"2021-06-
10T08:22:45.692862Z","src_ip":"192.168.250.147","session":"bb8b736d0154"}
```


Figure 37 Example of Honeypot log for Cowrie Honeypot

Honeypot LOG ETL is the component devoted to parsing and analysing the honeypot log files. Basically, it allows for extracting new threats and interfacing with a MISP instance to make available these new attack data in the TIP. The log files directory of each honeypot is synchronized by means of *rsync* with another one available on the Gateway Router. *TCP Dump Script* periodically stores the gathered networks flows from the Untrusted to the Trusted Network as PCAP files. In Figure 38 we show the traffic flow extraction performed by the Honeypot LOG ETL.

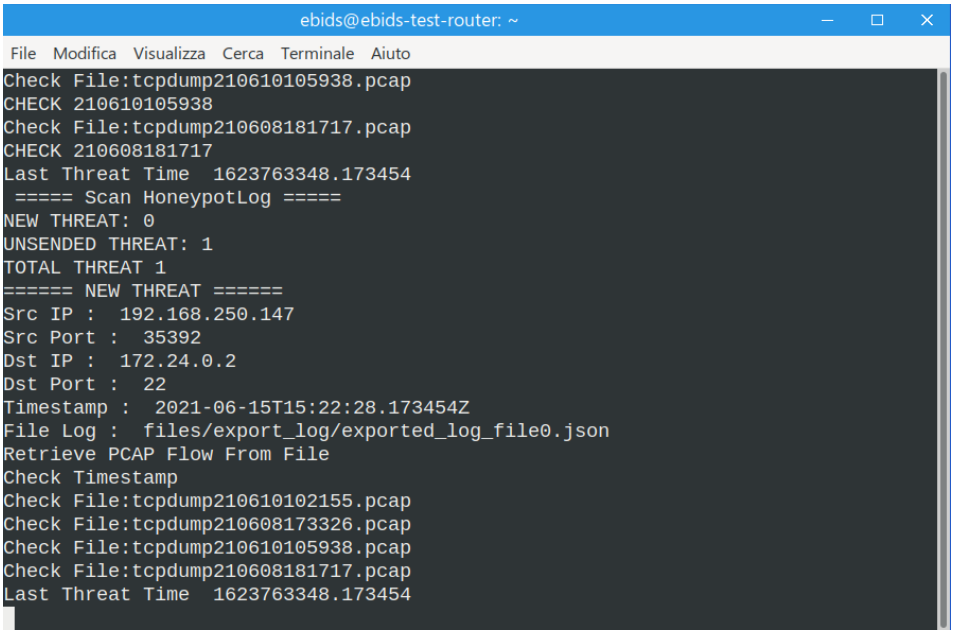

Figure 38 Honeypot LOG ETL in action

Basically, the *Honeypot LOG ETL* checks the occurrence of new data on the shared local directory deployed on the Gateway Router, then extracted network flow is associated with the threat. Finally, a new MISP Object is shared with the MISP by using the "Security Event Object" template. The honeypot log is stored in the `anomaly_details` field.

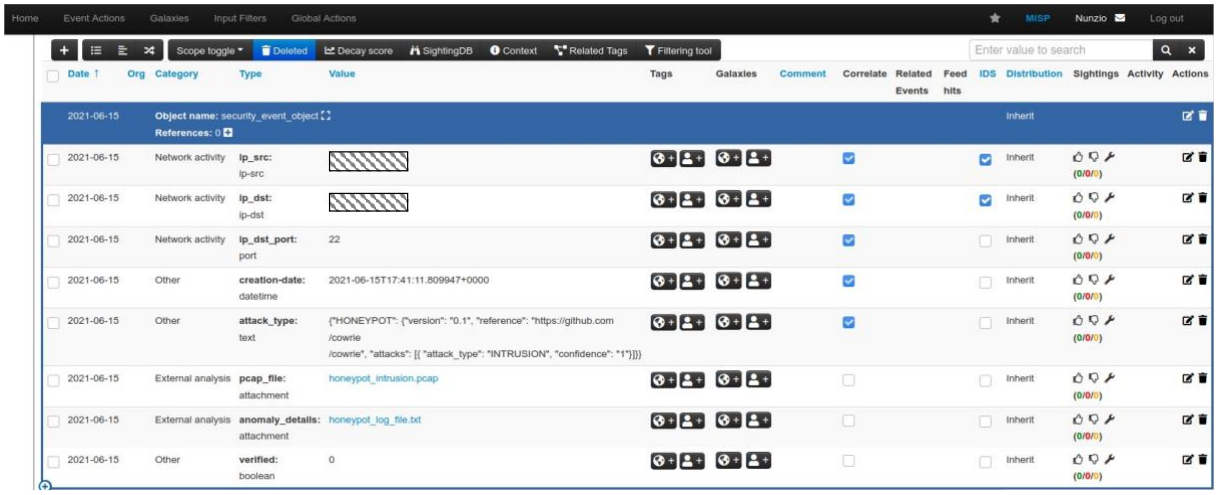

Figure 39 Honeypot data stored on MISP

As above-mentioned, the honeypot's PCAP shared with the MISP will be used as further positive examples (i.e., attacks) in the learning phase of IDS, as shown in Figure 39. In the following an example attack type attribute generated by Honey IDS is shown:

```
{
  "HONEYPOT": {
    "version": "0.1",
    "reference": "https://github.com/cowrie/cowrie",
    "attacks": [
      {
        "attack_type": "INTRUSION",
        "confidence": "1"
      }
    ]
  }
}
```

Figure 40 Example of attack type attribute for Honeypot Threat

A video demonstration of the scenario is available at:

https://github.com/massimo-guarascio/cs4e_ebids_asset

3.3 Scenario 3: Adaptive Deployment

3.3.1 Summary

Gathering relevant information on attack strategies and sharing precious IoCs to improve the security degree of the entities belonging to the MISP network is the main objective of this use case. In more details, this scenario aims at demonstrating how internal information gathered by means of a pool of honeypots can be used in the context of the security of an infrastructure.

Two assets collaborate to achieve this aim: (i) Briareos, a HIDS capable of launching Honeypots in any linux system and (ii) Roce, a system devoted to evaluating the risk of compromise of a network segment based on the information about known attacks on the software contained in a device. The information extracted by Briareos and data from a public APT database are combined by Roce and exploited to yield the probability of compromise of the monitored systems.

3.3.2 Assets Roles

3.3.2.1 Briareos

The gathering of threat knowledge is one of the most important and necessary tasks in today's cybersecurity. The increase of attacks led to changes in how to collect knowledge of (potential) threats and how we can take advantage of that knowledge. Briareos can be exploited in this context, by providing a specific paradigm for data collection and enrichment. Figure 41 shows the overview of the architecture.

The Malware Information Sharing Platform (MISP) (Wagner, et al. 2016) component will update their database with the Common Vulnerabilities and Exposures (CVE) entries (1). The Briareos Central Coordinator will act as a go-between managing the devices scanning history and honeypots. From time to time the devices will query the coordinator for new software to scan. That request is forwarded to MISP that will return a list of new entries - for both CVE and IoC (2). For each CVE, the coordinator parses the Common Platform Enumeration (CPE) list and sends it to each device in the network to check if it is installed on their system (3) and the device will provide feedback of any matches (4).

When a match is found, the Briareos Central Coordinator will select a random device and launch a honeypot with the same operating system and software version of the match (5). Inside the honeypot, an admin user can verify if the match is not a false positive and provide that information to the Briareos Central Coordinator (6).

From the honeypot logs (7), an administrator user can observe the malicious requests and can create, if possible, rules to be applied to the Briareos Host Intrusion Detection System (HIDS) of all devices (8).

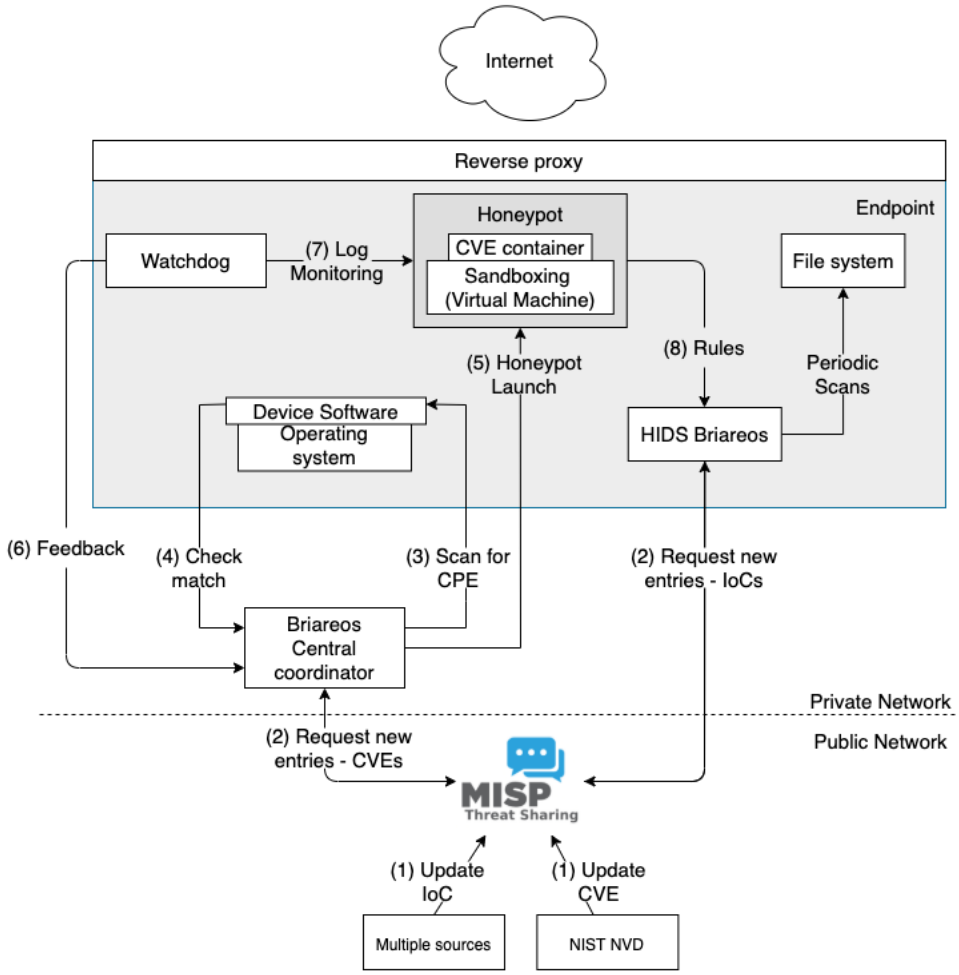

Figure 41. System's architecture of Briareos

Briareos Central Coordinator - Check CVE and Launch Honeypot. This component is the gateway between MISP and the devices in the network. It is responsible for checking if a device in our network has a vulnerable software version installed and to launch honeypots.

This parse of CVE is coordinated by the Briareos Central Coordinator. This coordinator machine will periodically ask MISP for updates that responds with a list of CVEs containing the corresponding CPEs. For each CVE, there is a list of software (in CPE syntax) that is vulnerable and requires patches or upgrades, so the coordinator parses the CPE list and sends it to each device in the network to check if it is installed on their system. If a match is found, then the coordinator will launch a honeypot instance with the same operating system and match software installed on a random device, not necessarily the same, with two main purposes: to verify if the match was not a false positive, given that CPE syntax and OS names have differences, and to verify if the vulnerability exposed by the CVE is being exploited. Honeypots can also be launched manually with CVE-ready environments to gather artifacts that can be later added to MISP. Once launched, the admin user can monitor the honeypot from the device's file system and retrieve the desired information.

Watchdog. The Watchdog is an agent installed on the host's system to enable users to access any file from the honeypot without having to access it. From the Watchdog, a user can monitor the logs from the vulnerable service running on the CVE container. With that information, it can analyze logs and parse malicious queries that can be manually added later to the IoC database in MISP. This creates the possibility in MISP of knowing the inside attacks viewed or received by the Honeypots allowing a

prioritization of the threats in a global overview. The Watchdog can be adapted to any sort of installation of service in the virtual machine.

Honeypot. The Honeypot instances are virtual machines with service(s) exposed to the Internet. The installation of those services is done directly on the file system or by using containers. The instances are prepared to run in x86-64 and arm architectures thus enabling (virtually) the launching of an instance on any device on the network. The Honeypot instance is not static and can be easily shifted between devices.

3.3.2.2 RoCe

Knowing the structure of a network, it is important for the network users and administrators to understand the risk of network compromise. Such information helps network administrators to plan their maintenance activities to prevent critical security breaches that could lead to, for example, denial of service or leakage of sensitive information that circulates inside the network.

RoCe aims at estimating the risk of compromise of a given network. For this, it needs to have the following inputs:

- List of software products (e.g. Office, Acrobat, etc.) installed in the network with their software version
- The current patching strategy implemented
- Any additional patching strategy that the company wants to evaluate for future deployments
- The information about known attacks or advanced persistent threats on the components used in the network

The database of APTs is collected and curated by UNITN, while the other information is retrieved from the Briareos asset. For example, the list of software products installed in the network with their software versions is retrieved by analysing the devices in the network. The patching strategies are identified by extracting the history of software upgrades (both from device logs and by tracking the network changes in time) performed by network administrators or devices users. Figure 42 summarises the integration of the RoCe and Briareos assets.

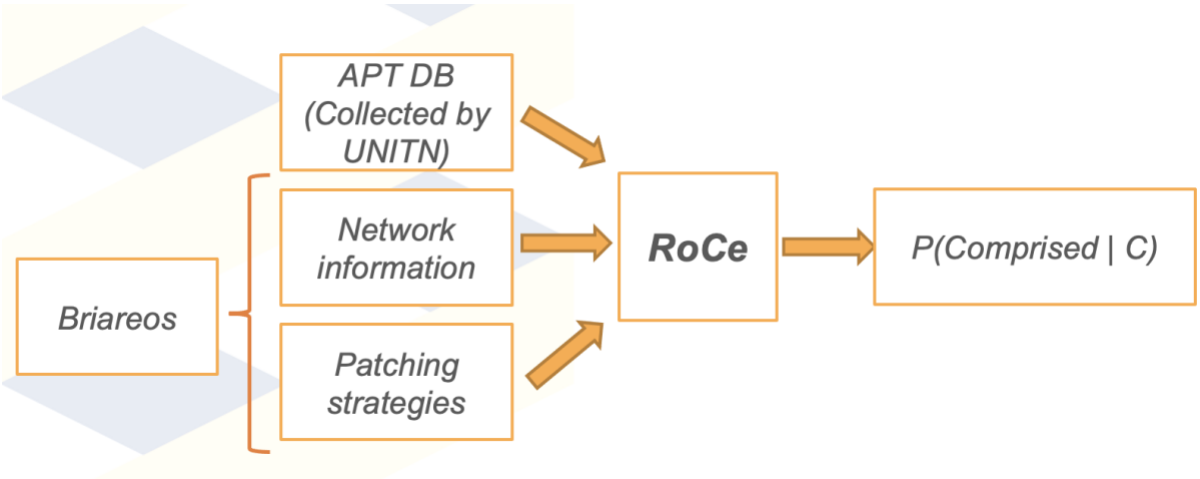

Figure 42: Integration of the RoCe and Briareos assets

3.3.3 Demonstrations

3.3.3.1 A Honeypot approach for the security of an infrastructure

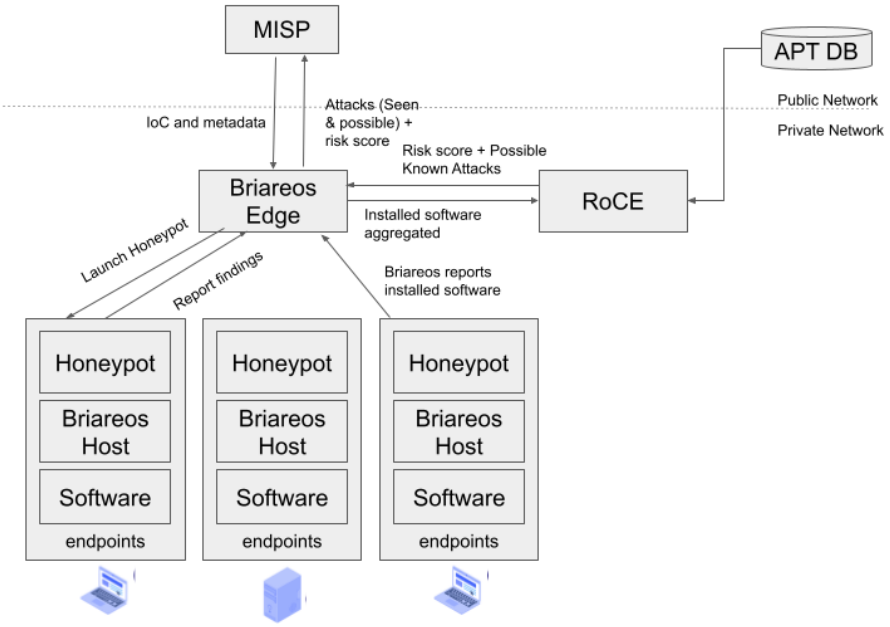

Figure 43: Interaction Workflow

Security Operation Centers (SOC) collect data from multiple sources on the Internet. A source example is MISP -threat intelligence platforms, by sharing and collecting indicators of compromise. These indicators of compromise can be collected from a wide set of services and mechanisms. An example of this is Honeypots that behave as a central point for data collections trying to see patterns or zero-days attacks on the network just by exposing the honeypots on the network.

Typically, the Honeypots run on a different segment of the network, focusing mainly on the scans that affect machines open to the Internet. In this context, we focus on the internal behavior of a network, with services that may only be available to internal users. This new type of integration allows the full integration of Briareos and RoCe, as illustrated in Figure 43.

Briareos will behave as a HIDS that is also capable of launching Honeypots on the device. For this, it assumes an isolated environment where it is possible to collect metrics and detect access to the services in the Honeypot. RoCe will rely on the Briareos to extract the list of software installed on the device. Then RoCe will map this list onto the dataset of known Advanced Persistent Threats (APT). As a result, RoCe will compute the risk score of each individual workstation and/or server inside of the network to be compromised by one of the known APTs.

In the end, three main functionalities will rise:

1. Briareos will perform the communication with MISP to receive IoC and block threats;
2. Based on the information collected from Briareos, RoCe calculates the score of risk of compromise for each individual workstation and/or server as the result of a known APT;
3. Based on CVE and the most relevant vulnerable software, Briareos launches Honeypots randomly on the network containing vulnerable software to test the security on the local network.

4 Summary and Outlook

We proposed a general framework of interaction and cooperation with threat intelligence services and provided three specific use cases where these services can cooperate. The partners contributed by providing tangible software assets, by integrating them into joint proof-of-concepts, and illustrating the practical feasibility of a modular cybersecurity platform able to provide key information about the status of a system to monitor. This deliverable documents 3 possible integration scenarios, culminating in a variety of videos, online demonstrators and scientific publications. Within these scenarios, we illustrate how such cooperation can (i) improve the performances of the threat prevention and detection systems and minimize the attack surface by strengthening the robustness of machine learning and deep learning models, making them more robust to new threats, false positives and lowering the time to threat detection; and (ii) enable more robust threat intelligence by allowing to better contextualize threat data and devise flexible strategies, methodologies and data formats for collaborative threat intelligence.

It is useful to consider the specific challenges raised in D3.3 and review how the approach proposed here addresses them.

1. The **amount of false positive threats must be reduced** to address alert fatigue with security professionals.
    - The solutions described in Scenario 2 can be used to reduce the false alarm rate by using the sharing protocol that allows different actors (either AI or humans) to validate threat evidence and mutually benefit from feedbacks provided by other peers. In addition, the interchangeability of ad-hoc solutions (as provided, e.g., by IntelFrame that provides a method of selecting more suitable learning classifiers based on different contexts) allows the underlying classifiers to improve the detection performance.
2. The **time to threat detection must be lowered**.
    - The use of the collaborative threat intelligence platform described in Scenario 2 allows the use of automated predictive models that reduce the average time to detect an intrusion.
3. Threat information **needs to be better contextualized using data from different information sources**.
    - Within scenario 2, events are enriched with further information. In particular, TIE evaluates and quantifies the impact of a security event within an organization's infrastructure. This security events are yielded and shared among interconnected institutions as well as external indicators of compromise received from open-source intelligence (OSINT) to improve contextualization of event. The data enrichment in HADES provides additional information about potential malware files detected by our platform. Reliable CTI Sharing asset acts allows for enriching the information about the trustworthiness of its source. By using some threat variables, including the behaviour of who has sent and created them, it yields a multi-dimensional score that informs about the trust of the shared information.
    - Within scenario 3, the assets Briareos and RoCe collaborate to enrich threat intelligence information with intelligence obtained by dynamically and adaptively deploying honeypots, and with a risk of compromise that is contextually dependent on the network infrastructure - including servers and workstations - as well the locally deployed services and applications. The vulnerabilities (CVE) are enriched with extended internal information in MISP. In particular, Briareos evaluates the internal network of organizations for vulnerable servers and actively deploys Honeypots inside of the internal network to enrich the information on MISP. It also quantifies the impact of a security event within an organization's infrastructure.

4. The **definition of new strategies, methodologies and formats to store and interpret the digital evidence** is crucial to improve the efficiency of the analysis.
    - A custom MISP object in JSON format is proposed. This template is flexible enough to satisfy the scenarios proposed in the demonstrator.
5. It is necessary to create **enabling technologies that can help boost trust among producers and consumers of threat intelligence information** and platforms to overcome the reluctance of sharing sensitive information with one another. In all cyberthreat analysis scenarios there is a need for **techniques that uphold the use of privacy enhancing techniques**.
    - Within scenario 1, we focus on sharing cyber threat intelligence in a confidential and privacy-preserving manner. In particular, the TATIS asset can be used to encrypt or privatize network related threat information provided by NIDS or HIDS solutions, using CP-ABE or other anonymization techniques for a more trustworthy sharing of threat intelligence. Also, the Privacy-Preserving CTI Sharing asset aims to carry out an anonymization process over sensitive information from threat events by applying PET mechanisms. This asset is complimentary to the cryptographic techniques, such as CP-ABE, that TATIS applies.
6. It is necessary to define **new solutions flexible enough to simplify the interpretation and understanding of the context** for different experts establishing **mechanisms to appropriately notify all the different actors**.
    - The proposed framework defines a platform for sharing, detecting, and analysing cyberthreats that is flexible and available for consultation form a variety of different actors. Moreover, the custom MISP object allows to share IoC and analysis results transparently with organizations external to the project.

In terms of next steps, this deliverable offers a foundation for further collaboration and research:

- The continuous arms race of attackers and defenders, with the former aiming to evade ML-based detection and the latter aiming to create more robust ML models.
- Deceiving adversaries in honeypots and honeynets to acquire more actionable insights.
- Standardization of taxonomies for ML-based assets and threats.
- More advanced threat intelligence services aimed at sharing ML models for threat detection and prevention.

List of resources

The list of resources can be accessed via the dedicated demonstration page, available at http://github.com/cs4ewp3t4 and are documented in the following.

Scenario 1: Sharing cyberthreat intelligence in a confidential and privacy-preserving manner:

- Online proof-of-concept demonstrators and repositories:
    - https://nuage.cs.kuleuven.be/ [KUL's MISP instance]
    - https://nuage.cs.kuleuven.be/tatis/ [integration with KUL's MISP instance]
    - https://ciel.cs.kuleuven.be/tatis/ [integration with CNR's MISP instance]
    - http://155.54.95.184:8082/anonymizer/AnonymizerAPI.html [PP-CTI anonymizer service, integrated with TATIS]
- Videos:
    - https://people.cs.kuleuven.be/~davy.preuveneers/tatis.mp4

- Scientific dissemination: (Preuveneers and Joosen 2020), (Preuveneers and Joosen 2021), (Preuveneers, Joosen and Bernal Bernabe, et al. 2020).

Scenario 2: Enriching the information on detected threats via TDS cooperation and gathered by means of honeypot instances:

- Online proof-of-concept demonstrators and repositories:
  - https://misp1.icar.cnr.it/ [CNR MISP instance]
- Videos:
  - https://github.com/massimo-guarascio/cs4e_ebids_asset
  - https://tinyurl.com/tie-atos
- Scientific dissemination: (Folino, et al. 2021), (Guarascio, et al. 2021)

Scenario 3: Adaptive deployment:

- Videos:
  - https://dcc.fc.up.pt/~jresende/video/briareos
- Scientific dissemination: (Pinto Bastos Martins 2020), (Dinis da Silva e Barbosa 2020)

5 References

Dandurand, L., and O. S. Serrano. 2013. "Towards improved cyber security information sharing." *35th International Conference on Cyber Conflict (CYCON 2013).* 1-16.

Dinis da Silva e Barbosa, Mário. 2020. *JBriareos: A Secure and Scalable Distributed Intrusion Detection System.* Master Thesis, https://sigarra.up.pt/fcup/pt/pub_geral.show_file?pi_doc_id=274710.

Folino, F., G. Folino, M. Guarascio, F.S. Pisani, and L. Pontieri. 2021. "On learning effective ensembles of deep neural networks for intrusion detection." *Information Fusion* 48-69.

Gonzalez-Granadillo, G., M. Faiella, I. Medeiros, R. Azevedo, and S. Gonzalez-Zarzosa. 2019. "Enhancing Information Sharing and Visualization Capabilities in Security Data Analytic Platforms." *49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W).* 1-8.

Guarascio, M., N. Cassavia, F.S. Pisani, and G. Manco. 2021. "Boosting Cyber Threat Intelligence via Collaborative Intrusion Detection." *Under Review.*

Johnson, C. S., M. L. Badger, D. Waltermire, J. Snyder, and C. Skorupka. 2016. "NIST Guide to cyber threat information sharing."

Lashkari, A. H., G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani. 2017. "Characterization of tor traffic using time based features." *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - Volume 1: ICISSP*. 253-262.

Pinto Bastos Martins, Maria Inês. 2020. *Anomaly Detection in Cybersecurity*. Mater Thesis, https://sigarra.up.pt/fcup/pt/pub_geral.show_file?pi_doc_id=275358.

Preuveneers, D., and W. Joosen. 2021. "Sharing Machine Learning Models as Indicators of Compromise for Cyber Threat Intelligence." *Journal of Cybersecurity and Privacy* 140-163.

—. 2020. "TATIS: Trustworthy APIs for Threat Intelligence Sharing with UMA and CP-ABE." *Foundations and Practice of Security. 12th International Symposium, FPS 2019. Revised Selected Papers.* Springer International Publishing.

Preuveneers, D., W. Joosen, J. Bernal Bernabe, and A. Skarmeta. 2020. "Distributed Security Framework for Reliable Threat Intelligence Sharing." *Security and Communication Networks.*

Samarati, P. 2001. «Protecting respondents identities in microdata release.» *IEEE transactions on Knowledge and Data Engineering* 13 (6): 1010--1027.

Wagner, C., A. Dulaunoy, G. Wagener, and A. Iklody. 2016. "The Design and Implementation of a Collaborative Threat Intelligence Sharing Platform." *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security (WISCS '16).* Association for Computing Machinery. 49-56.

Zibak, A., and A. Simpson. 2019. "Cyber threat information sharing: Perceived benefits and barriers." *Proceedings of the 14th International Conference on Availability, Reliability and Security, ARES'19.* Association for Computing Machinery.