# D4.1 - SPHINX Cross-layer anomaly detection framework v1

## WP4 – SPHINX Toolkits

**Version: 1.00**

SPHINX

A Universal Cyber Security Toolkit for Health-Care Industry

### *Disclaimer*

Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

### *Copyright message*

**© SPHINX Consortium, 2019**

This deliverable contains original unpublished work except where clearly indicated otherwise.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

### *Document information*

| Grant Agreement Number | 826183 | Acronym | SPHINX |
|---|---|---|---|
| **Full Title** | A Universal Cyber Security Toolkit for Health-Care Industry | | |
| **Topic** | SU-TDS-02-2018 Toolkit for assessing and reducing cyber risks in hospitals and care centres to protect privacy/data/infrastructures | | |
| **Funding scheme** | RIA - Research and Innovation action | | |
| **Start Date** | 1st January 2019 | **Duration** | 36 months |
| **Project URL** | http://sphinx-project.eu/ | | |
| **EU Project Officer** | Reza RAZAVI (CNECT/H/03) | | |
| **Project Coordinator** | National Technical University of Athens - NTUA | | |
| **Deliverable** | D4.1 SPHINX Cross-layer anomaly detection framework v1 | | |
| **Work Package** | WP4 – SPHINX Toolkits | | |
| **Date of Delivery** | **Contractual** M20 | **Actual** | M20 |
| **Nature** | R - Report | **Dissemination Level** | P - Public |
| **Lead Beneficiary** | SIMAVI | | |
| **Responsible Authors** | Radu Popescu, Dana Oniga | **Email** | radu.popescu@siveco.ro, dana.oniga@siveco.ro |
| | | **Phone** | |
| **Reviewer(s):** | PDMFC, ICOM | | |
| **Keywords** | Anomaly Detection, Data Traffic Monitoring | | |

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

*Document History*

| Version | Issue Date | Stage | Changes | Contributor |
|---|---|---|---|---|
| 0.10 | 10/02/2020 | Draft | ToC | Mircea Vasile (SIMAVI) |
| 0.20 | 27/07/2020 | Draft | Content | Daniela Condrache (SIMAVI) |
| 0.30 | 27/07/2020 | Draft | Content | Catalin Danila (SIMAVI) |
| 0.40 | 7/08/2020 | Draft | Content | Radu Popescu (SIMAVI) |
| 0.50 | 26/08/2020 | Draft | Internal Review | Stylianos Karagiannis (PDMFC), Ilias Lamprinos (ICOM) |
| 0.60 | 27/08/2020 | Draft | Address Reviewers' Comments | Radu Popescu (SIMAVI) |
| 0.70 | 28/08/2020 | Pre-Final | Address Reviewers' Comments | Dana Oniga (SIMAVI) |
| 0.80 | 28/08/2020 | Pre - Final | Quality Control | George Doukas (NTUA) , Michael Kontoulis (NTUA) |
| 1.00 | 28/08/2020 | Final | Final | Christos Ntanos (NTUA) |

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

# Executive Summary

The document D4.1 Cross-Layer Anomaly Detection Framework v1 presents the research and development activities that were made to design and build the first version of two of the SPHINX Toolkit components: Data Traffic Monitoring (DTM) and Anomaly Detection (AD).

These components are included in the **Automated Cyber Security Risk Assessment** block**,** one of the main high-level SPHINX Architecture building blocks, which deals with advanced and automated tools to assess the level of cyber security of a given environment (e.g., healthcare information technology operational environment)

This deliverable contributes toward reaching SPHINX project milestone MS5: First set of SPHINX services & modules prototype completed - First Integrate approach at the end of M20. The development of both components (DTM and AD) is at the end of their first iteration (M13-M20). Both components underwent a systematic research, design and development work. The components are continuously growing and maturing. This report presents the overall development achievements in the first iteration.

First chapter is the introduction to this report, giving more details about its purpose, structure and showing its relations to others WPs and Tasks.

Chapter 2 General Overview explains the investigation of concepts on which Data Traffic Monitoring (DTM) and Anomaly Detection (AD) components are based on

Chapter 3 describes DTM Component, starting with its scope, functionalities and design principles. More details are given on DTM's background, technical construction and relations with other SPHINX Toolkit Components.

Chapter 4 is dedicated to the AD Component, explaining its functionalities, architecture, background, technical construction and relations with other SPHINX Toolkit Components.

The last Chapter concludes that the two components are complementary, DTM detecting known threats based on signatures, while AD is designed to detect unknown threats by creating profiles of normal behaviour and searching for network traffic events that are considered anomalous because they do not fit the profiles.

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

# Contents



*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

# Table of Figures



*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

# List of Abbreviations

ABD - Anomaly-based Detection

AD - Anomaly Detection

API - Application Programming Interface

DTM - Data Traffic Monitoring

FDCE - Forensic Data Collection Engine

HP - Honeypot

ID - Interactive Dashboards

IDS -Intrusion Detection System

IPS - Intrusion Prevention System

ML - Machine Learning

NIDS - Network Intrusion Detection System

NSM - Network Security Monitoring

PCAP - Packet Capture

REST - REpresentational State Transfer

RCRA - Real-time Cyber Risk Assessment

SB - Sandbox

SBD - Signature-based Detection

SIEM - Security Information and Event Management

SPA - Stateful Protocol Analysis

WP - Work Package

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

# 1 Introduction

## 1.1 Purpose & Scope

The purpose of this document is to present the systematic work done under the task *T4.1 SPHINX Anomaly Detection and User Profiling (WP4: SPHINX Toolkits)* between months M13 to M20 of SPHINX project. This deliverable will cover the activities of research, design and development performed during the first iteration of development for two of the SPHINX Components: Data Traffic Monitoring (DTM) and Anomaly Detection (AD). The two components will work together in order to provide a framework for anomaly detection.

## 1.2 Structure of the deliverable

This deliverable is structured in four chapters. Chapter 1 is the introduction and presents the rationale for writing this document and its relations with other WPs & Tasks. Chapter 2 contains a general overview of the subject of anomaly detection in network traffic; Chapter 3 and Chapter 4 describe Data Traffic Monitoring and Anomaly Detection Components, covering their role in the SPHINX Ecosystem, technical and architectural details and integrations with other SPHINX components. Chapter 4 presents a short summary and general conclusion of this document.

## 1.3 Relation to other WPs & Tasks

The development effort described by this report is based on deliverables from WP2 Conceptualisation, Use Cases and System Architecture: D2.4 Use Cases Definition and Requirements Document v1, D2.5 Requirements and Guidelines v1, D2.6: SPHINX Architecture v2. Design, development and implementation principles were used based on recommendations from WP6: SPHINX Common Integration Platform & Incremental Strategy.

The output of DTM and AD components will be made available to other components from WP4 SPHINX Toolkits and WP5 Analysis and Decision Making.

AD and DTM will be integrated, validated and tested in the context of WP6: SPHINX Common Integration Platform & Incremental Strategy, during tasks T6.4 System integration execution that will address the actual integration of outcomes from WP4 and WP5 and T6.5 Testing of Integrated SPHINX platform.

The second iteration of development of AD and DTM components will include feedback received after integration testing and feedback received from end users during the pilots within WP7- Technology Validation Pilots and Privacy assessment. Task T7.2 System functional testing and validation is aimed to verify that the deployed pilots and the components meet the functional, operational and technical requirements as described in other WPs and will gather the evaluation results of the validation pilot release (M25-M27) and the second validation pilot release (M32-M34).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

# 2 General Overview

The Cross-Layer Anomaly Detection Framework is a system responsible with the detection of malicious activities by monitoring and analysing network traffic. These types of systems are called Intrusion Detection Systems (IDS). It is designed to be modular, configurable, and extensible in order to be easily adaptable to hospitals with various types of network infrastructures. The framework must support networks with different types of topologies, sizes and volume traffic.

In order to respond to these requirements a systematic investigation was done into specific literature, sources and materials, as described below.

The literature on the subject of network monitoring and intrusion detection classifies the detection methodologies as follows [1]:

- **Signature-based Detection (SBD)** - A signature is a pattern or string that corresponds to a known attack or threat. SBD is the process to compare patterns against captured network traffic for recognizing possible intrusions. Because of using the knowledge accumulated by specific attacks and system vulnerabilities, SBD is also known as Knowledge-based Detection or Misuse Detection.
- **Anomaly-based Detection (ABD)** - An anomaly is a deviation from a profile. A profile represents the normal or expected behaviours derived from monitoring regular activities, network connections, hosts or users over a period of time. Profiles can be either static or dynamic and developed for many attributes, e.g., failed login attempts, processor usage, the count of e-mails sent, etc. Then, ABD compares normal profiles with observed events to recognize significant attacks. ABD is also called Behaviour-based Detection in some articles. Some ABD examples are attempted break-in, masquerading, penetration by legitimate user, Denial-of-Service (DOS), Trojan horse, etc. SBD and ABD are complementary methods, because the former concerns certain attacks/threats and the latter focuses on unknown attacks.
- **Stateful Protocol Analysis (SPA)** - The stateful in SPA indicates that the Intrusion Detection System (IDS) could know and trace the protocol states (e.g., pairing requests with replies). Thought SPA process looks like ABDs, they are essentially different. ABD adopts preloaded network or host-specific profiles, whereas SPA depends on vendor-developed generic profiles to specific protocols. Generally, the network protocol models in SPA are based originally on protocol standards from international standard organizations, e.g., IETF. SPA is also known as Specification-based Detection.

The methodologies studied and presented above are implemented in various tools. The Cross-Layer Anomaly Detection Framework of SPHINX is based on open source tools, libraries in order to obtain a solution easily adaptable and extendable based on the specific requirements of the clients.

We focused mainly on Signature-based and Anomaly-based detection methodologies. By using multiple methodologies we have created a hybrid IDS [2] to provide more extensive and accurate detection. We have developed two components responsible with the implementation of these methodologies:

- Data Traffic Monitoring (DTM) – implements the signature-based detection approach
- Anomaly Detection (AD) – implements the anomaly-based detection approach

The main functions of these components are:

- Detection of intrusions in the monitored network by using signature-based detection and anomaly-based detection that complement each other.
- Reporting of the detected threats and alerting.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

- The users responsible for the IT security have access to the detected threats and alerts through the Interactive Dashboard, another component of the SPHINX Toolkit.
- AD and DTM publish alerts to a Kafka broker in order to make them available to other components of the SPHINX Ecosystem.
- Exposing API endpoints that enable access to data and functionalities. Other SPHINX components or third-party applications can request access to detailed traffic data following an alert.

The AD and DTM components will be described in the following sections of the document.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

# 3 Overview of Data Traffic Monitoring

## 3.1 Scope of Data Traffic Monitoring

Data Traffic Monitoring is a SPHINX component responsible with threat identification by monitoring the network traffic and applying signature-based detection analysis. It monitors all the packets traversing the network and compares them against a database of attack signatures or attributes of known malicious threats.

DTM is a Network Intrusion Detection System (NIDS) optimized to work in the SPHINX Ecosystem by communicating with other SPHINX components and exposing alerts and relevant statistics to the users.

The main functionalities of the Data Traffic Monitoring (DTM) component are:

- capturing traffic from multiple protocols;
- analysing packets and files in different formats;
- identifying traffic information for every user and source;
- highlighting unusual communication/activity according to the rules and filters defined
- identifying new assets on the network.

## 3.2 Design Principles

The Data Traffic Monitoring component has to capture relevant network traffic in order for its analysis techniques to be successful. If the network is organized in subnets, DTM must be able to do its analyses on the local traffic from the subnets in order to detect potential threats that don't generate external traffic (for example, a malware on a compromised device that does a port scan attack in its subnet). DTM is designed to support agents that will be deployed at strategic points within the network. The agents are controlled from a central management DTM instance.

Another consideration in designing the DTM component is the ability to easily modify and extend its capabilities in order to adapt to specific details of the infrastructures where it is deployed. That is why DTM is designed to make it simple to integrate new tools in the solution.

DTM integrates the following tools:

- **Tshark** – this is a network protocol analyser. Tshark offers the ability to capture packet data from a live network or allows reading packets from a previously saved capture file. It has a powerful package filtering support and protocol dissection capabilities. DTM and Tshark can be used together for investigations after threat detection or to develop complex custom detection analyses procedures. [2]
- **Suricata** – this is an open source, mature, fast and robust network threat detection engine [3]. It has powerful and extensive rules and signature language for network traffic inspection. There are many prebuilt rules available that cover known attacks and vulnerabilities.

Persistent data, like alerts, statistics, agent configuration data and operation data is stored in a PostgreSQL database.

Alerts and statistic information are made available to other application in two ways:

- by publishing messages to Kafka
- by exposing the data as REST services

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

The figure 3.1 describes the DTM component with the details presented above.

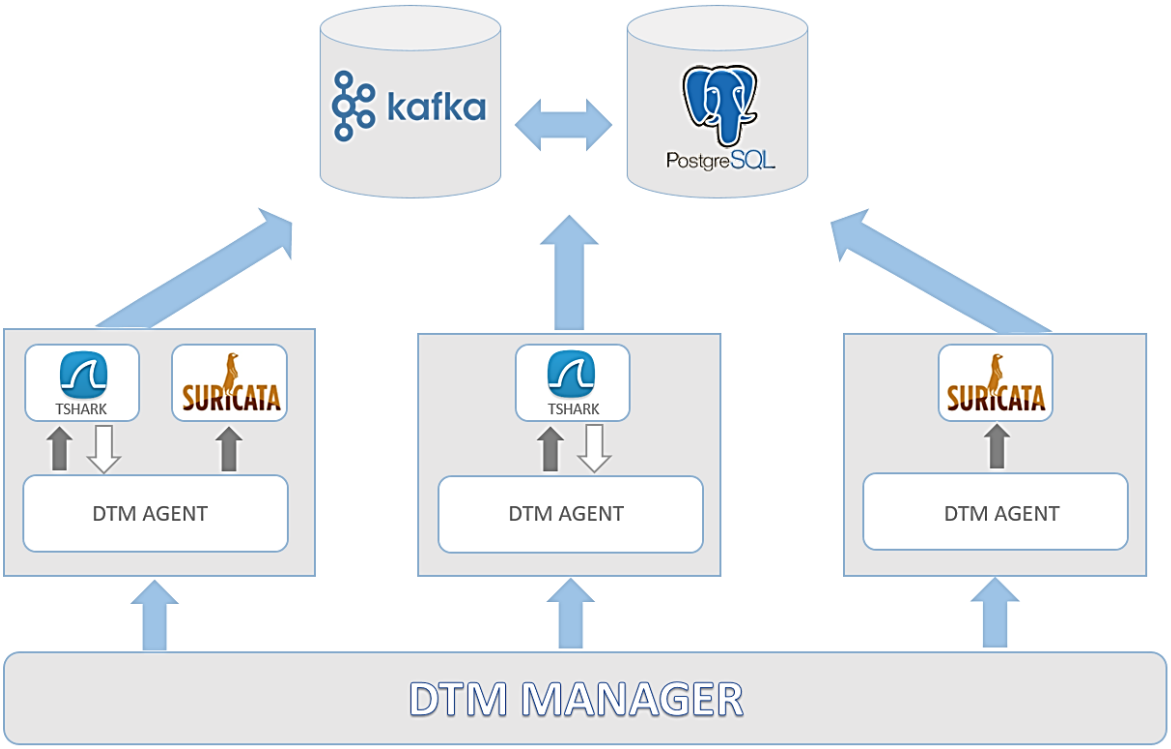



***Figure 3.1 SPHINX DTM Architecture***

The elements represented in figure 3.1 are:

- DTM Manager – this is the component that contains features like the agent administration and control, statistics.
- DTM Agent – the agents are deployed on the network in order to capture the relevant traffic. Each agent can control tools for network traffic capture and analysis like Tshark and Suricata. The communication between the manager and the agents is based on REST webservices.
- Kafka – the message broker used for internal and external communication
- PostgreSql – is used for storing configurations and statistics.

## 3.3 Human factor

DTM can be configured to search for a vast number of possible threats. For example, Suricata has many predefined rules and signatures regarding protocols at various layers in OSI model (like protocols TCP, UPD that correspond to transport layer in OSI model or HTTP, FTP that are application layer in OSI model). The evaluation of the rules consumes processing power, so it is important to deactivate the rules that are not relevant in the environment that is monitored. It is the user that must decide which are the rules that must be active.

DTM raises alerts when it detects threats. It is the responsibility of the user to analyse the alerts and fix the root cause for the problem. For example, following an alert the user can take actions like:

- update the software in order to close vulnerabilities
- educate the operators in order to prevent email scam campaign

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

- update the rules used to evaluate the traffic if they are no longer correct due to changes in the network traffic characteristics

## 3.4 Technical Details

DTM component is based on Tshark and Suricata IDS as proven open source tools used in network monitoring, network security investigations and threat detection and prevention. DTM Manager and Agents are created with Spring Boot framework and Java version 14. DTM Manager also contains a web front-end that is created with React JavaScript library for UI components and is running on Node.js runtime.

DTM deployment scenarios depend on the infrastructure of the network DTM will monitor. A possible deployment is represented in the following figure.

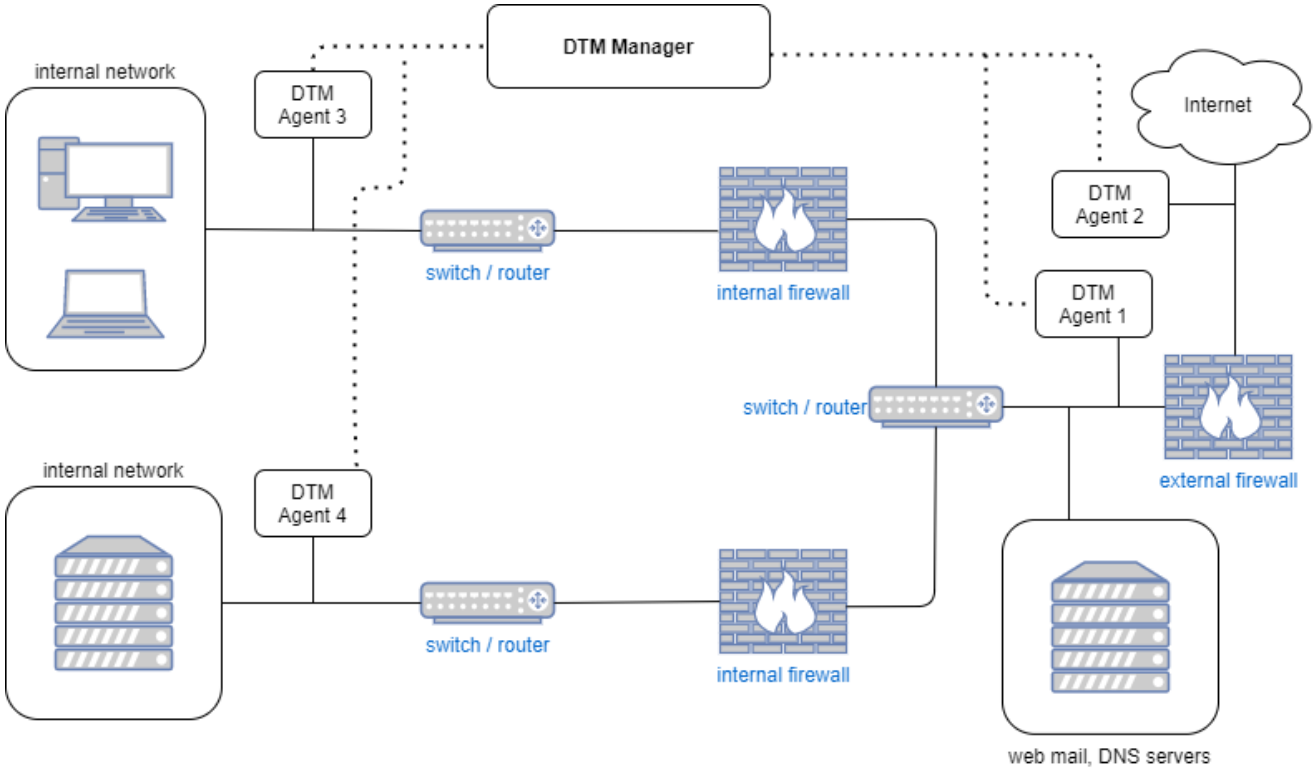



***Figure 3.2 SPHINX DTM Deployment scenario***

In this deployment scenario, DTM agents are deployed at various points in the infrastructure:

- DTM agent 1 – this agent is deployed behind the external firewall. It will analyse external in / out traffic.
- DTM agent 2 – it is also possible to deploy an agent in front of the external firewall. The network traffic that this agent analyses has more noise because it will see traffic that is not blocked by the firewall. But the alerts raised by this agent can be useful to configure the firewall in order by block suspicious traffic as early as possible.
- DTM agents 3, 4 – these agents monitor subnets. Monitoring the internal traffic of the subnets is important in order to detect threats that escape the agents deployed around the external firewall. For example, it is possible to infect a computer in a subnet with a malware using an external drive. DTM will detect the malware's network activity.

The logical components of DTM are described in the following sections.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

### 3.4.1 DTM Manager

The DTM Manager is the component responsible with the administration and control of the DTM agents. It allows defining new agents, choosing between integration with Tshark and Suricata and configuring specific information for Tshark or Suricata.

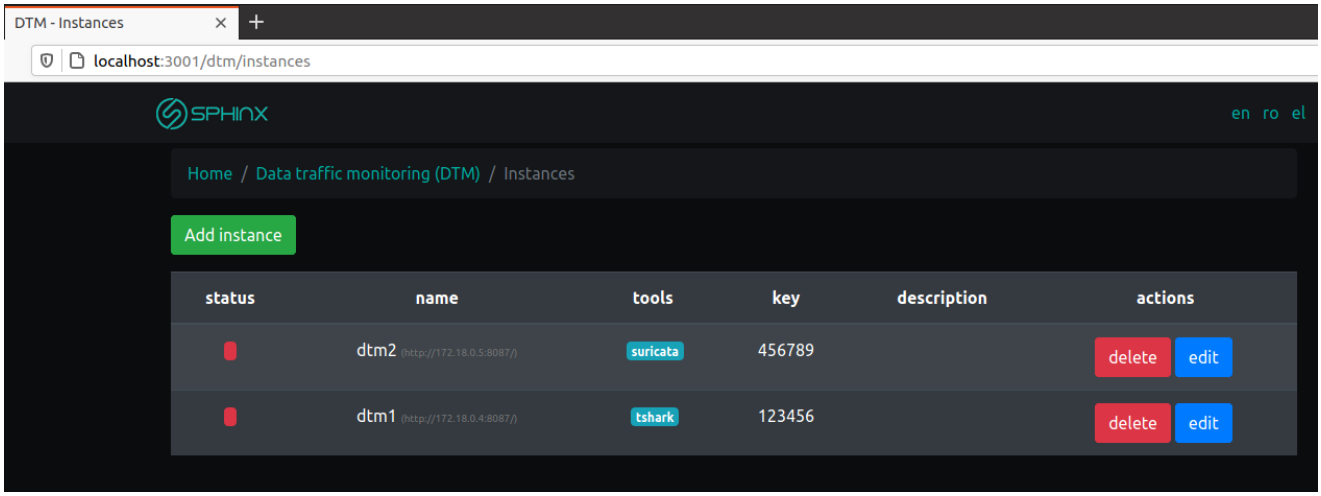



***Figure 3.3 SPHINX DTM Agents list***

In figure 3.3 is visible the list of configured agents. New agents can be added using the “Add instance” button. Existing agents can be deleted or modified with the buttons in the “actions” column.

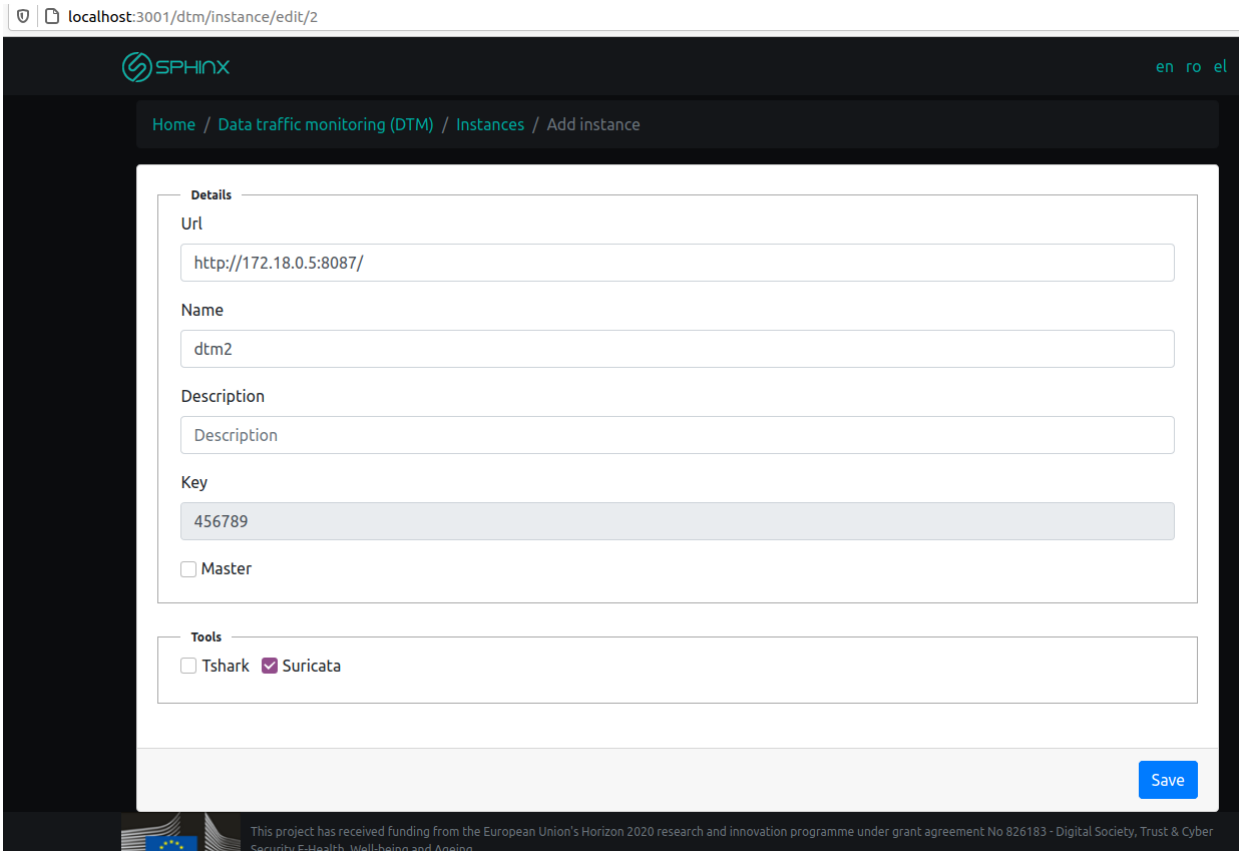



***Figure 3.4 SPHINX DTM Add Agent page***

In figure 3.4 is the page for configuring an agent. This page opens after using the “Add instance” button above. Here it is possible to configure details like the url where the agent is accessible, a name and a key. There can be only one agent with the property Master set. This agent plays the role of DTM Manager. In the Tools section, it is possible to select the tools deployed with the agent (Tshark, Suricata).

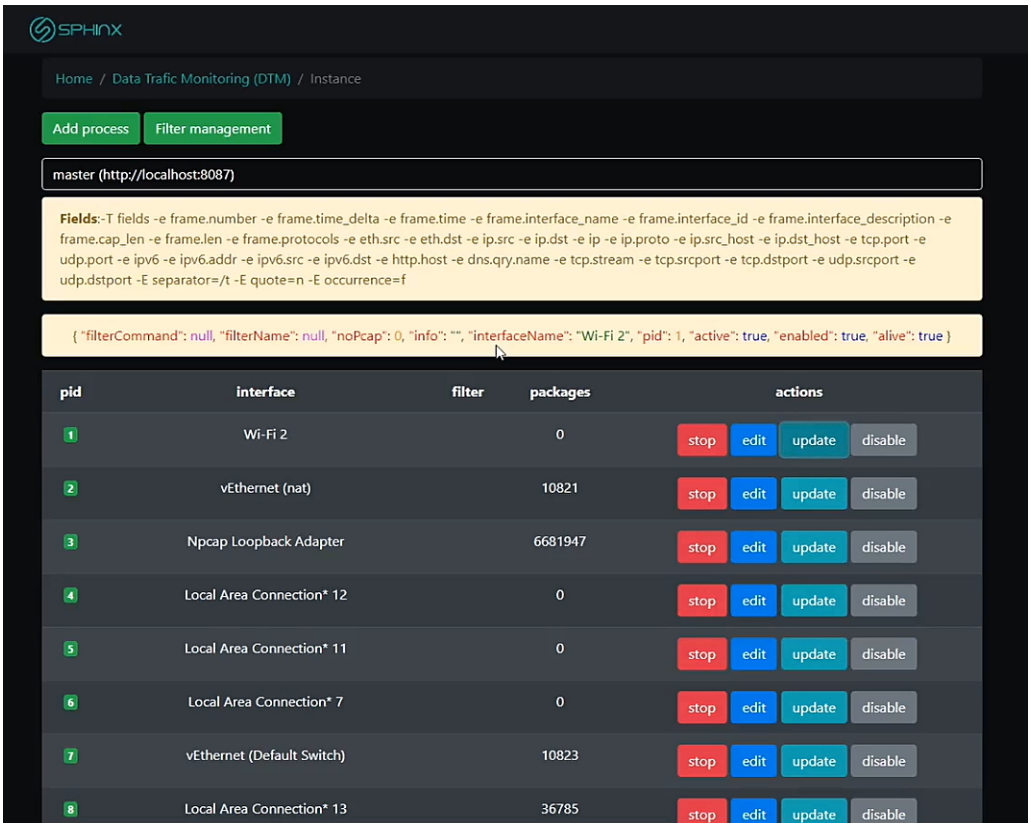



***Figure 3.5 SPHINX DTM Instance details screen***

In figure 3.5 is the administration page for an agent working with Tshark. It is possible to configure on what interfaces the agent will capture traffic and what filter is used for filtering traffic.

The DTM manager also contains various statistics about the network environment as represented in figure 3.6.

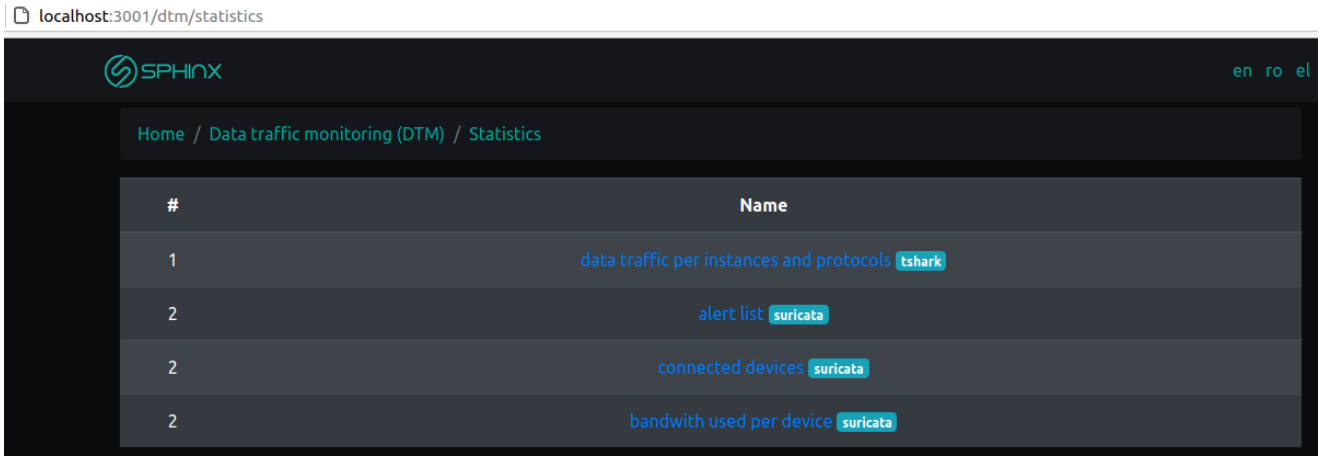



***Figure 3.6 SPHINX DTM Statistics***

As examples of statistics that are created by DTM:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

- Data traffic per instance and protocols
- Alerts
- List of connected devices
- Bandwidth used per device

For example, accessing “Data traffic per instance and protocols” brings the page in figure 3.7:

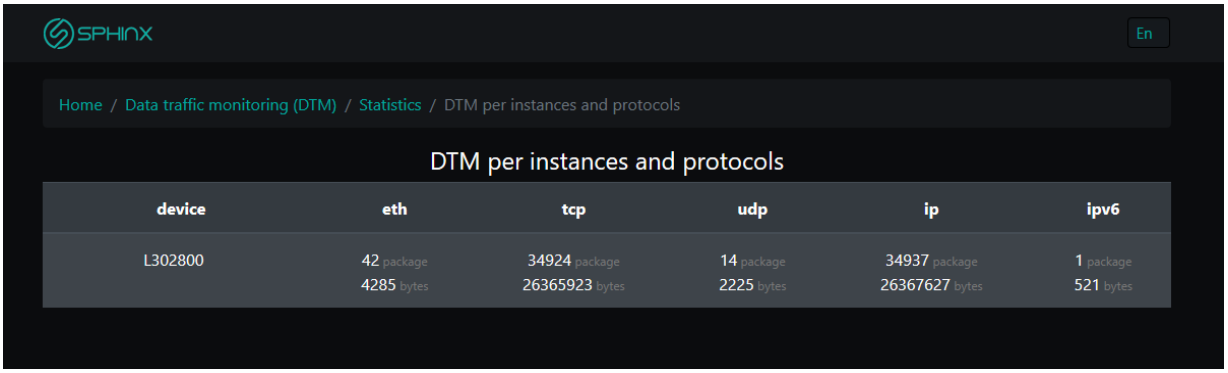



***Figure 3.7 SPHINX DTM per instances and protocols***

### 3.4.2 DTM Agents

DTM Agents are deployed in strategic points of the network infrastructure in order to see the relevant network traffic. For example, the agents will be installed on proxy servers or on computers that have access to monitoring port of routers or switches.

The agents are controlled by the DTM Manager.

### 3.4.3 Tshark

Tshark is a network protocol analyser. It captures packet data from a live network, or reads packets from a previously saved capture file, either printing a decoded form of those packets to the standard output or writing the packets to a file. Tshark's native capture file format is pcapng format, which is also the format used by wireshark and various other tools.

Packet capturing is performed with the pcap library. That library supports specifying a filter expression; packets that do not match that filter are discarded. The syntax of a capture filter is defined by the pcap library.

Tshark also supports read filters, which allow selecting which packets are to be decoded or written to a file. Read filters are very powerful; more fields are filterable in Tshark than in other protocol analyser, and the syntax used to create filters is richer.

### 3.4.4 Suricata

Suricata is a free and open source, mature, fast and robust network threat detection engine. The Suricata engine is capable of real time intrusion detection (IDS), inline intrusion prevention (IPS), network security monitoring (NSM) and offline pcap processing. Suricata inspects the network traffic using a powerful and extensive rules and signature language and has powerful Lua scripting support for detection of complex threats. With standard input and output formats like YAML and JSON, integrations with tools like existing SIEMs, Splunk, Logstash/Elasticsearch, Kibana and databases become effortless.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

### 3.4.5 Asset discovery

DTM monitors the network traffic in order to identify new devices. Unknown devices are a potential security threat. DTM raises alerts regarding the new devices identified. The list of unknown and known devices is accessible in the component's web application (figure 3.8).

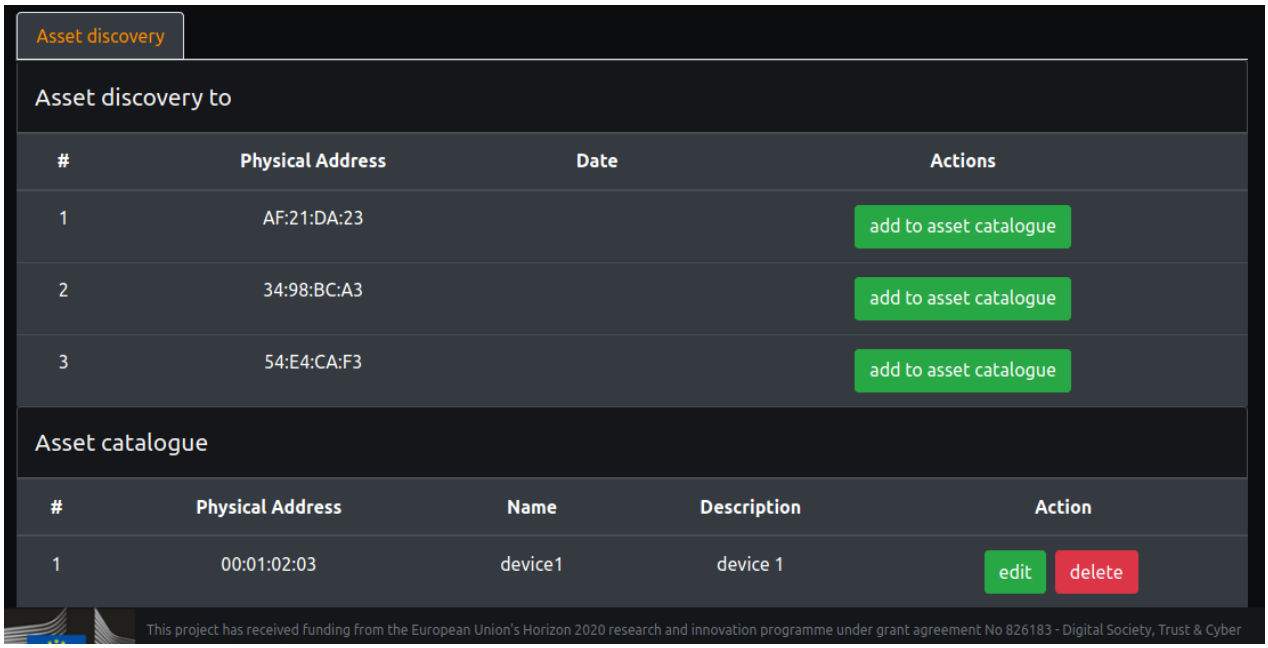



***Figure 3.7 SPHINX DTM asset discovery***

The alerts are visible in the Interactive Dashboard component and are also published to Kafka in order to be used by other SPHINX Components, for example Vulnerability Assessment as a Service (VAaaS) component.

### 3.4.6 Database

DTM component uses for data persistence a PostgreSQL database.

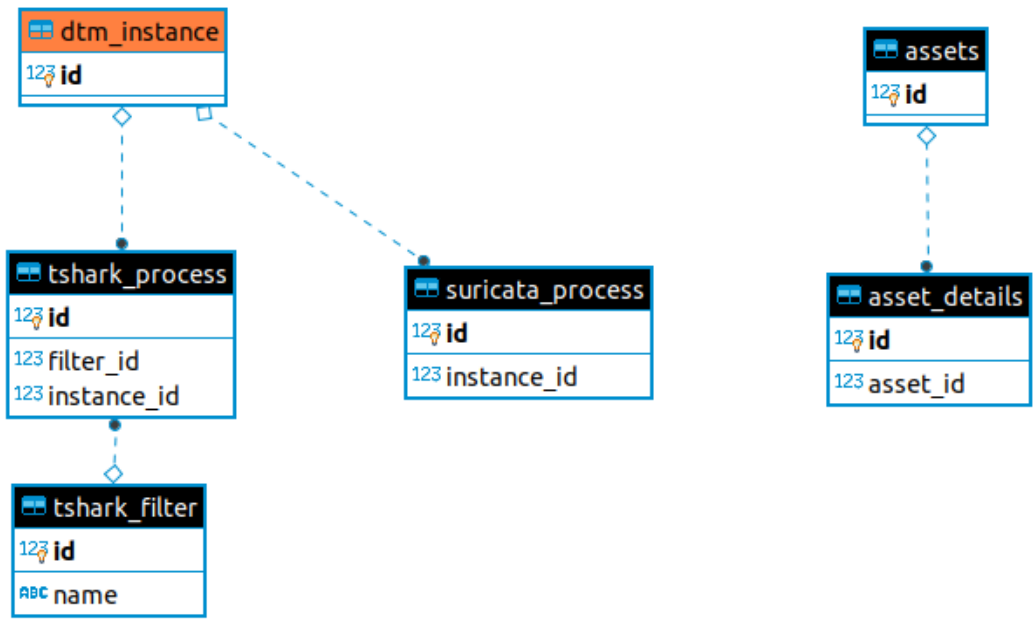



***Figure 3.9 SPHINX DTM database diagram***

A short description of the tables:

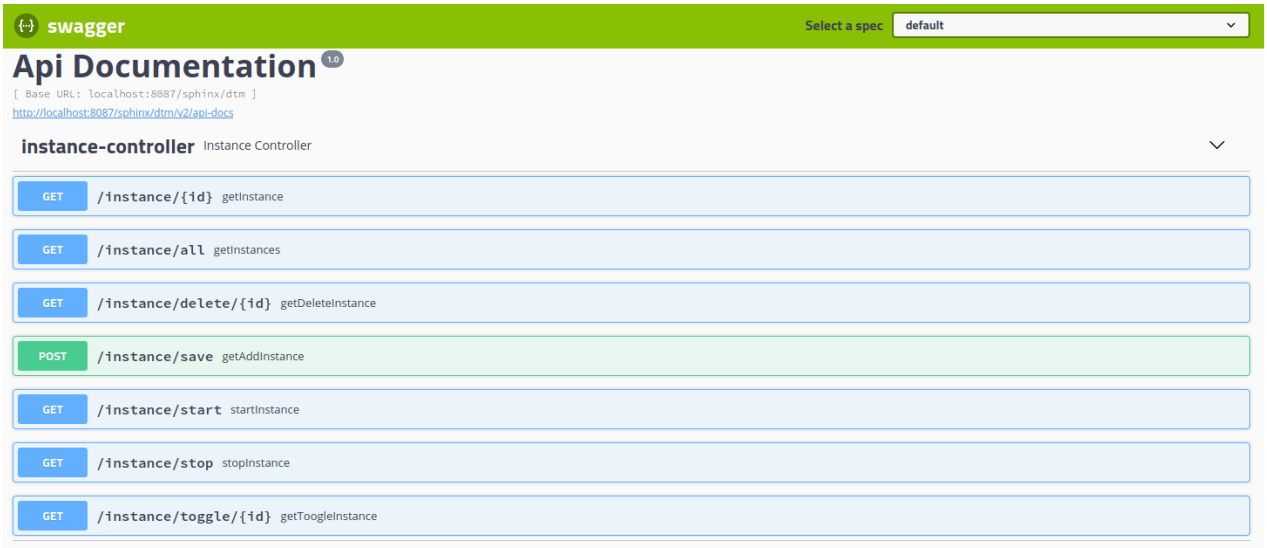
- dtm_instance – the list of agents.
- Tshark_process – the list of Tshark processes configured on agents. An agent can be integrated with Tshark, Suricata or both.
- Tshark_filter – the filters used by Tshark processes for network monitoring.
- suricata_process – the list of suricata processes configured on agents.
- assets – the list of devices discovered in the network
- asset_details – information about the discovered devices

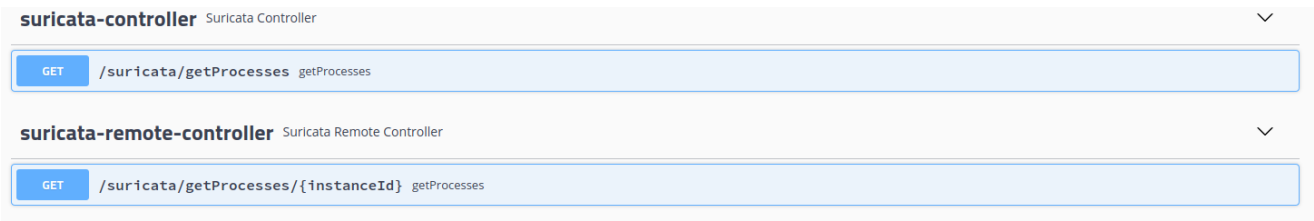
### 3.4.7 API

DTM component uses REST services both for internal usage and for integration with external components.

Internally, the web front-end reads and writes data to the java backend. Also, the DTM agents expose REST services used by the DTM Manager.

The following figures present the API used by DTM as generated by swagger.




This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

**tshark-controller** Tshark Controller

| | |
|---|---|
| GET | /tshark/disable/{pid} disable |
| GET | /tshark/enable/{pid} enable |
| GET | /tshark/getFields getFields |
| GET | /tshark/getInterfaces getInterfaces |
| GET | /tshark/getProcesses getProcesses |
| POST | /tshark/process/save saveProcess |
| POST | /tshark/process/update updateProcess |
| GET | /tshark/start/{pid} start |
| GET | /tshark/status/{pid} status |
| GET | /tshark/stop/{pid} stop |
| GET | /tshark/up isUp |

**tshark-filter-controller** Tshark Filter Controller

| | |
|---|---|
| GET | /tshark/filter/{id} getFilter |
| POST | /tshark/filter/add add |
| GET | /tshark/filter/all getFilters |
| GET | /tshark/filter/delete/{id} delete |
| POST | /tshark/filter/save save |

**tshark-remote-controller** Tshark Remote Controller

| | |
|---|---|
| GET | /tshark/disable/{pid}/{instanceId} disable |
| GET | /tshark/enable/{pid}/{instanceId} enable |
| GET | /tshark/getInterfaces/{instanceId} getInterfaces |
| GET | /tshark/getProcesses/{instanceId} getProcesses |
| GET | /tshark/metric/{name}/{instanceId} getMetric |
| POST | /tshark/process/save/{instanceId} saveProcess |
| POST | /tshark/process/update/{instanceId} updateProcess |
| GET | /tshark/start/{pid}/{instanceId} start |
| GET | /tshark/status/{pid}/{instanceId} status |
| GET | /tshark/stop/{pid}/{instanceId} stop |

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

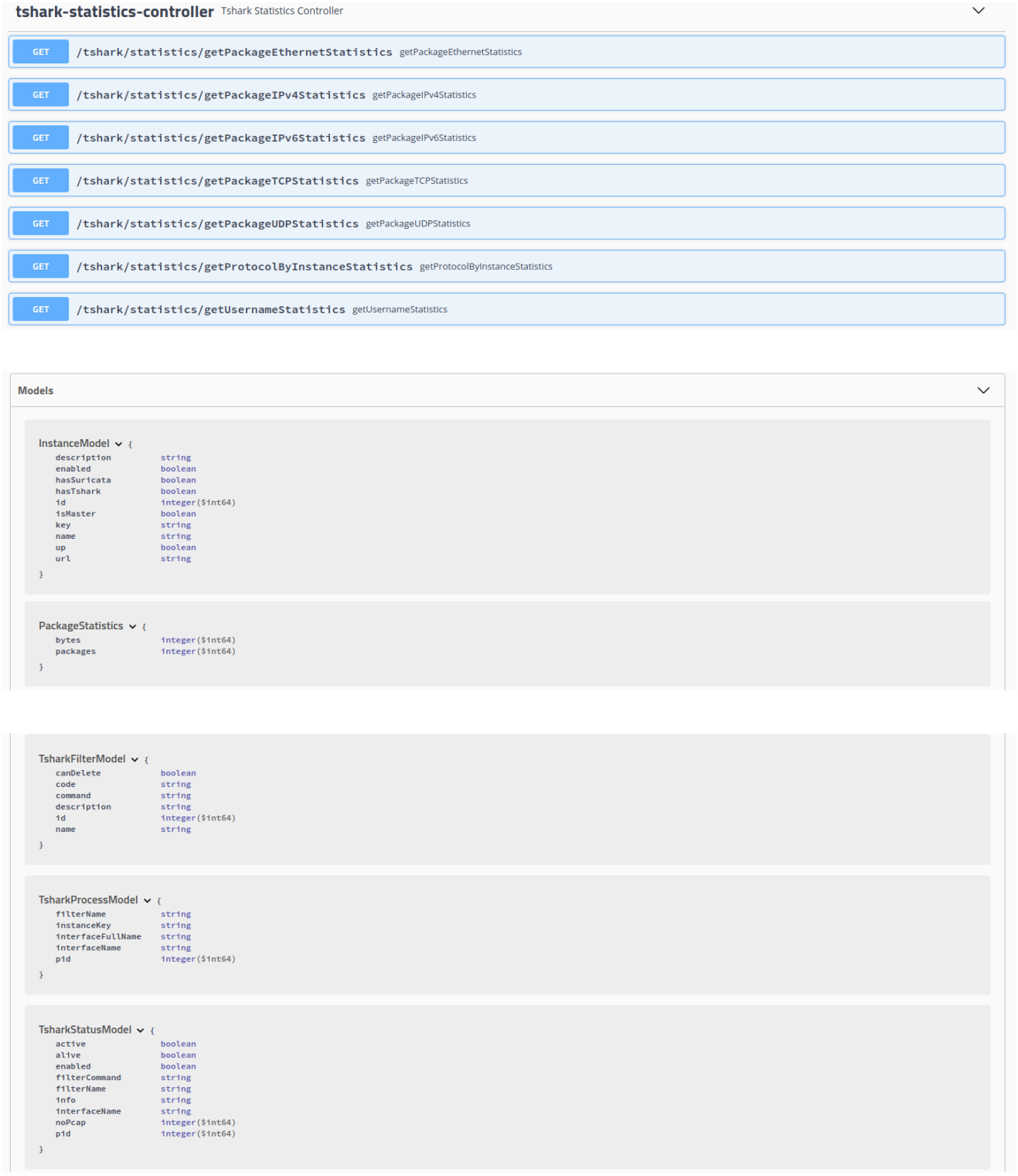



***Figure 3.8 SPHINX DTM API***

### 3.4.8 Kafka

DTM uses Kafka to asynchronously send information. In the figure 3.11 there are examples of Kafka topics where DTM publishes messages. The topic dtm-metric contains statistics; dtm-package contains summaries of the captured packages. The number of topics will increase as the component is further developed.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

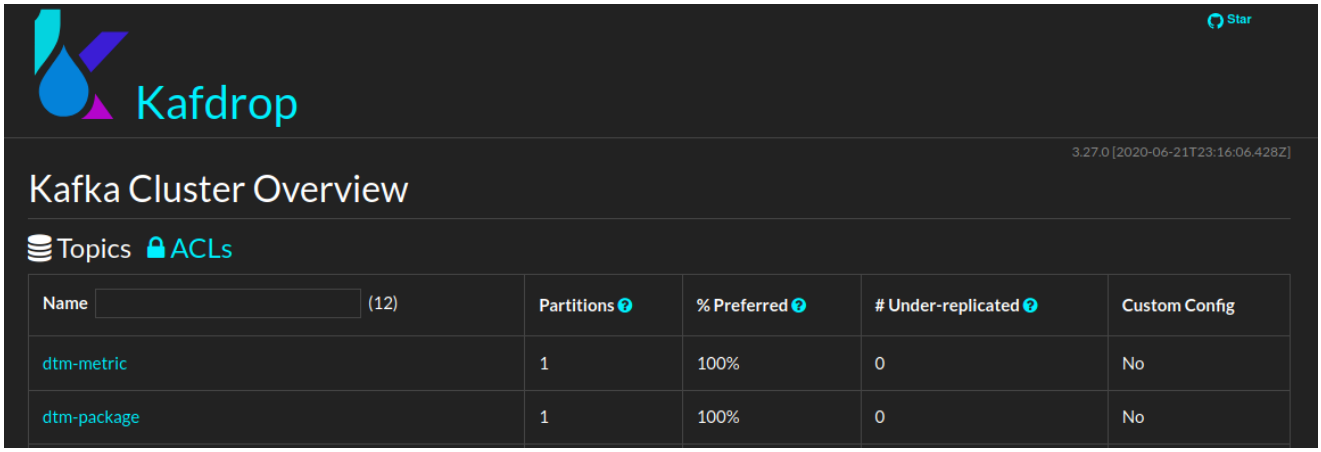



***Figure 3.9 SPHINX DTM Topics in Kafka***

## 3.5 Background

Data Traffic Monitoring component, part of the SPHINX Universal Cyber Security Toolkit, is targeted to for Healthcare institutions of different sizes, with network infrastructures of different complexities, covering one or more locations, in one or more towns. This leads to a variety of deployment scenarios and has the potential to be a complex solution that needs a large team of security experts to install, configure and administer the component.

In order to keep the complexity under control our approach is to create a tool that uses agents that are centrally managed and that transmit the data collected in a central location for visualisation and processing.

The agents are based on open source tools Tshark and Suricata. These are state of the art tools, well known and respected. They complement each other in the endeavour of network threat detection, investigation and exploratory testing of the network infrastructure.

Suricata is a IDS / IPS able to detect known threats, policy violations, malicious behaviour and anomalies. It is a multi threaded application, highly scalable capable of inspecting multi-gigabit traffic. It supports automatic protocol detection on any port.

Tshark, as a protocol analyser with powerful filtering support, is intended to be used as a investigatory tool after Suricata signals suspect traffic in the network. Tshark also supports a wider range of protocols at different layers in the OSI model.

## 3.6 Data Traffic Monitoring in Sphinx

Data Traffic Monitoring component is a support component for other components in the SPHINX ecosystem.

The relationships between DTM and the other SPHINX components are represented in the following image.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

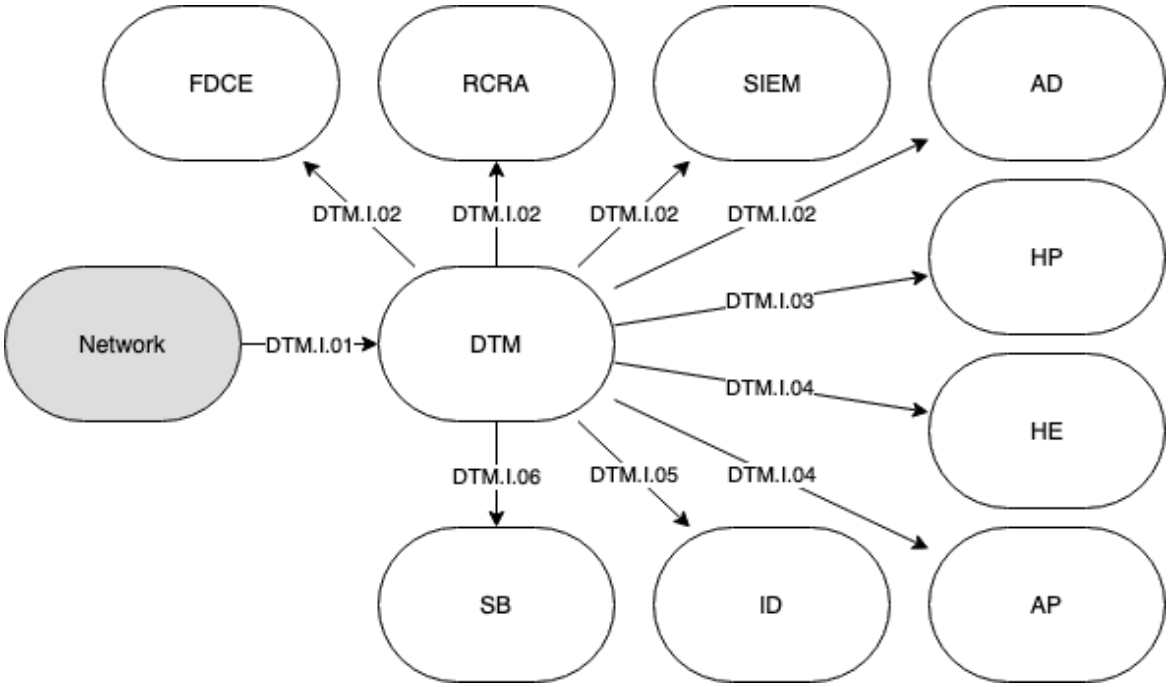



***Figure 3.10 SPHINX DTM Collaboration Diagram***

The DTM supports the following interactions with the other components:

- DTM sends abnormal and suspicious traffic activity to the Anomaly Detection (AD), the Forensic Data Collection Engine (FDCE), the Real-time Cyber Risk Assessment (RCRA) and the Security Information and Event Management (SIEM) components, including packets and files on traffic data and unusual activities concerning users and their connections.
- DTM sends abnormal and suspicious traffic data (data files and packets) to the Honeypot (HP) component to further detect and analyse possible attacks.
- DTM sends statistical information concerning collected data traffic (e.g., number of connected devices and connected users, data access type, bandwidth used per device and per user) to the Interactive Dashboards (ID) component.
- DTM sends traffic information, including information about connected assets (devices), to the Sandbox (SB) component, in order to support the complete mapping of the IT infrastructure. This information is used specifically for intrusion detection and alerts on Denial of Service attacks.

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.*

# 4 Overview of Anomaly Detection

## 4.1 Scope of Anomaly Detection

Anomaly Detection is a SPHINX component that uses a different approach than the DTM component to threat discovery. It analyses network activity and classifies it as either normal or anomalous. Instead of using signatures as a basis for classification, AD builds profiles for normal behaviours and uses data mining and machine learning algorithms in order to identify outliers that are reported as alerts.

AD does not use the raw network data. AD uses as input the logs generated by Data Traffic Monitoring component. These logs describe the network activity in high-level terms. For example, they can contain:

- TCP connections
- HTTP sessions with details like URIS, headers, MIME types, server responses
- DNS requests with replies
- SMTP sessions

DTM discovers only threats that are in its signature database. That means it can detect only known threats. The AD component creates a baseline for normal network/device/user behaviour. This approach of AD component allows it to detect new and unknown threats by monitoring the outlier activity that departs from baseline.

The main functionalities of this component are:

- detection of ecosystem disturbances;
- implement a set of rules based on the characteristics of previous system events, user activities and incidents;
- provide an alert engine to raise notifications.

## 4.2 Design Principles

The design of the AD component is based on the following considerations:

- SPHINX ecosystem is a complex system, with components built using different programming language and technologies. As part of this ecosystem, AD must be able to interoperate with the other SPHINX components. This is achieved in two ways:
  - by using a messaging system. The information is published in topics. Other SPHINX components subscribe to the topic of interest in order to get access to data.
  - by exposing RESTful web services.
- AD should be scalable in order to be usable with networks with both low traffic and high traffic volume. Because of this AD is based on big data tools and algorithms optimized for distributed execution.
- AD should be able to discover unknown threats, thus complementing the DTM component. In order to be able to discover unknown threats, AD uses statistics and machine learning in order to analyse the network activity, create profiles corresponding to normal activities and detect anomalous activities reported to these profiles.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

- AD should be configurable and extensible. Depending on the network environment where the component is deployed it is possible that not all the analyses implemented in AD are necessary. AD should permit to enable or disable supported analyses. Also, AD should allow to easily adding new analyses.

The following figure is a high-level representation of AD logical architecture and collaboration with other SPHINX Components.

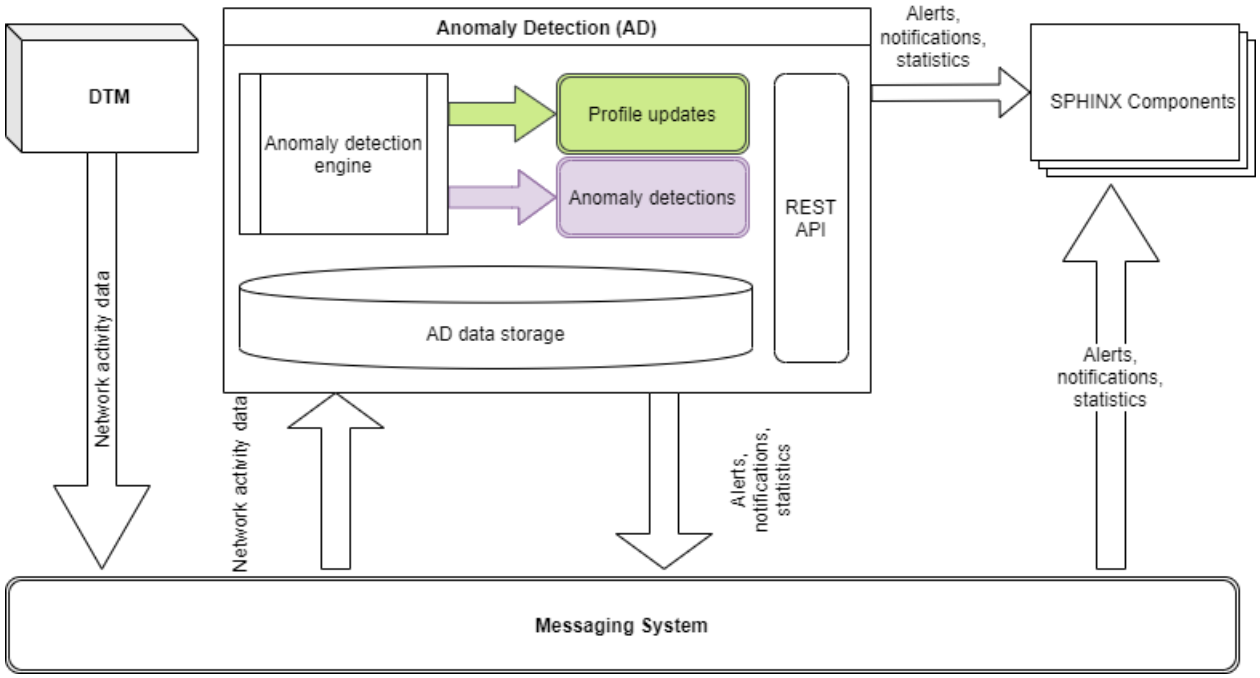



***Figure 4.11 SPHINX AD High Level Logical Architecture***

The internal logical architecture of the AD component is represented in the following figure:

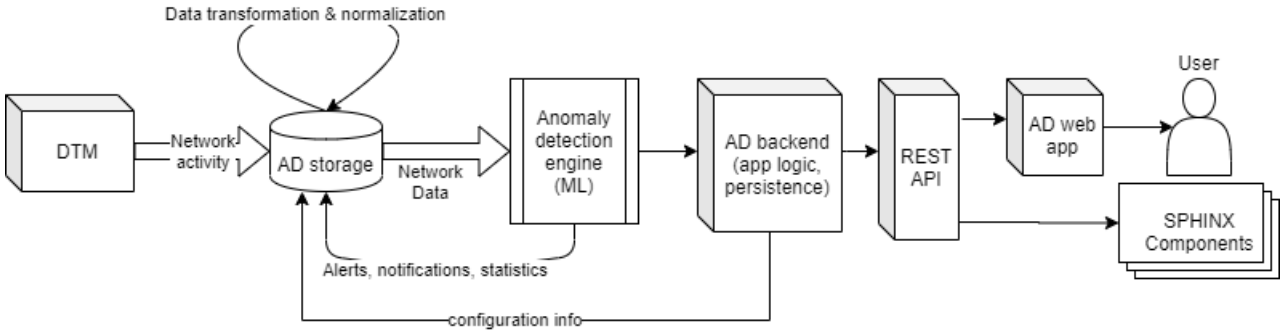



***Figure 4.12 SPHINX AD Internal Logical Architecture***

AD ingests the data about network activity published by the DTM component. The data is stored internally in a storage service capable of handling large data volumes. The anomaly detection engine applies statistical and ML algorithms in order to create a baseline of normal activity and detect anomalies. The backend contains application and persistence logic and exposes a REST API used both by the web subcomponent and other SPHINX components.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

## 4.3 Human Factor

For the AD component similar considerations apply as for the DTM component. AD issues alerts when it finds potential threats. It is the responsibility of the user to analyse the alerts and fix the root cause for the problem. For example, following an alert the user can take actions like:

- update the software in order to close vulnerabilities
- educate the operators in order to prevent email scam campaign
- update the rules used to evaluate the traffic if they are no longer correct due to changes in the network traffic characteristics

## 4.4 Technical Details

The tools used in the AD component are represented in the following figure:

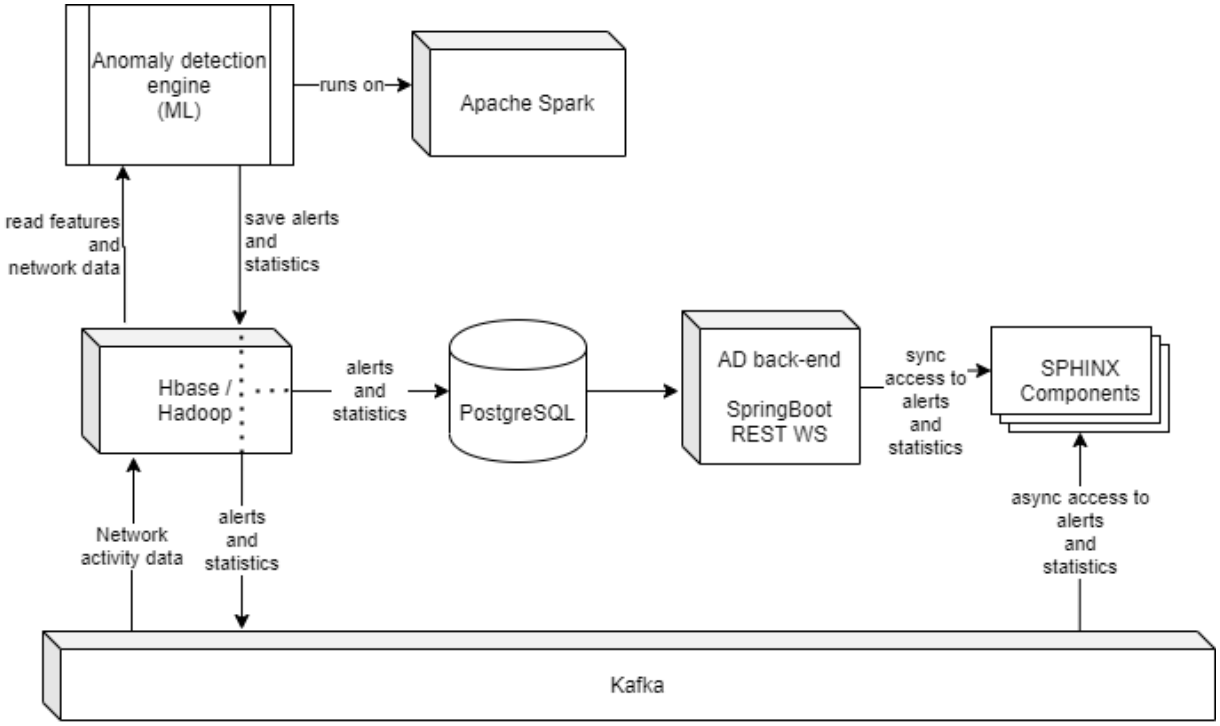



***Figure 4.13 AD back-end & front-end tools***

In order to handle a potentially high volume of network traffic, a solution based on Hbase, Hadoop and Spark was chosen. These tools are designed to support large datasets and to scale easily. Hadoop and Hbase offer distributed storage capabilities. Hbase is used because it offers real-time read/write capabilities on top of Hadoop. Spark supports performing parallel processing over distributed datasets and has a machine learning library that supports a rich set of algorithms.

A short description of each tool follows.

### 4.4.1 Hadoop

Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures [4].

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

The core of Apache Hadoop consists of a storage part, known as Hadoop Distributed File System (HDFS), and a processing part which is a MapReduce programming model. Hadoop splits files into large blocks and distributes them across nodes in a cluster. It then transfers packaged code into nodes to process the data in parallel. This approach takes advantage of data locality, where nodes manipulate the data they have access to. This allows the dataset to be processed faster and more efficiently than it would be in a more conventional supercomputer architecture that relies on a parallel file system where computation and data are distributed via high-speed networking.

### 4.4.2 Hbase

HBase is an open-source non-relational distributed database, developed as part of Apache Software Foundation's Apache Hadoop project and runs on top of HDFS (Hadoop Distributed File System) or Alluxio, providing Bigtable-like capabilities for Hadoop. It provides a fault-tolerant way of storing large quantities of sparse data [5].

HBase features compression, in-memory operation, and Bloom filters on a per-column basis. Tables in HBase can serve as the input and output for MapReduce jobs run in Hadoop. HBase is a column-oriented key-value data store and has been widely adopted because of its lineage with Hadoop and HDFS. HBase runs on top of HDFS and is well-suited for faster read and write operations on large datasets with high throughput and low input/output latency. HBase system is designed to scale linearly [5].

### 4.4.3 Spark

Apache Spark is an open-source distributed general-purpose cluster-computing framework. Spark provides an interface for programming entire clusters with implicit data parallelism and fault tolerance [6].

Apache Spark has its architectural foundation in the resilient distributed dataset (RDD), a read-only multiset of data items distributed over a cluster of machines that is maintained in a fault-tolerant way. The Dataframe API was released as an abstraction on top of the RDD, followed by the Dataset API [6].

Spark facilitates the implementation of both iterative algorithms, which visit their data set multiple times in a loop, and interactive/exploratory data analysis, i.e., the repeated database-style querying of data. The latency of such applications may be reduced by several orders of magnitude compared to Apache Hadoop MapReduce implementation. Among the class of iterative algorithms are the training algorithms for machine learning systems, which formed the initial impetus for developing Apache Spark [6].

Spark contains MLlib, a scalable machine learning library. At a high level, it provides tools such as [7]:

- ML Algorithms: common learning algorithms such as classification, regression, clustering, and collaborative filtering
- Featurization: feature extraction, transformation, dimensionality reduction, and selection
- Pipelines: tools for constructing, evaluating, and tuning ML Pipelines
- Persistence: saving and load algorithms, models, and Pipelines
- Utilities: linear algebra, statistics, data handling, etc.

### 4.4.4 Anomaly detection engine

The Anomaly Detection Engine consists of applications that run on Apache Spark. It uses MLlib from Apache Spark to execute summarization and threat detection tasks on the data describing network activity received from the DTM component.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

The results are saved in HBase database for future reference and are published in Kafka in order to be available for interested subscribers.

The network traffic data generated by DTM + Suricata consists of events specific to supported protocols, like DNS, HTTP, FTP, TLS etc. This information is used to create profiles with protocol specific information. Threat detection tasks are run for each relevant protocol.

The main approach used in Anomaly Detection Engine is to use a clustering technique in order to partition the network events in several cluster and to identify the outlier events. The features used are protocol specific. Spark ML library provides support for streaming k-means clustering that is useful when data arrive in a stream like the network events from DTM component. This approach permits to estimate clusters dynamically, updating them as new data arrive. This permits the update of profiles as the user behaviour changes in time.

### 4.4.5 AD backend & frontend

The AD back-end is created with Spring Boot framework and Java version 14. The front-end is created with React JavaScript library for UI components and is running on Node.js runtime. Configuration data and various statistics are stored in PostgreSQL database. This is represented in the following figure:

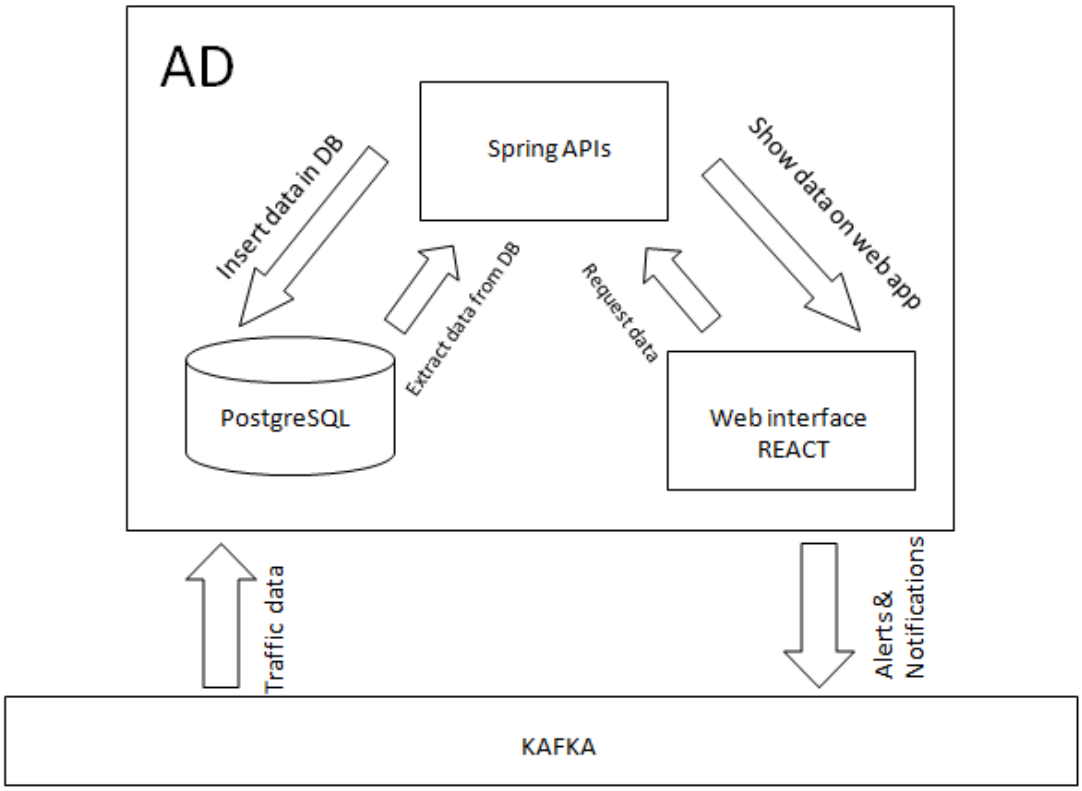



***Figure 4.14 AD back-end & front-end tools***

The Anomaly Detection (AD) front-end is divided into three subcomponents, namely: AD Components, Alert and Statistics (Figure 4.17) as represented in the figure below:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

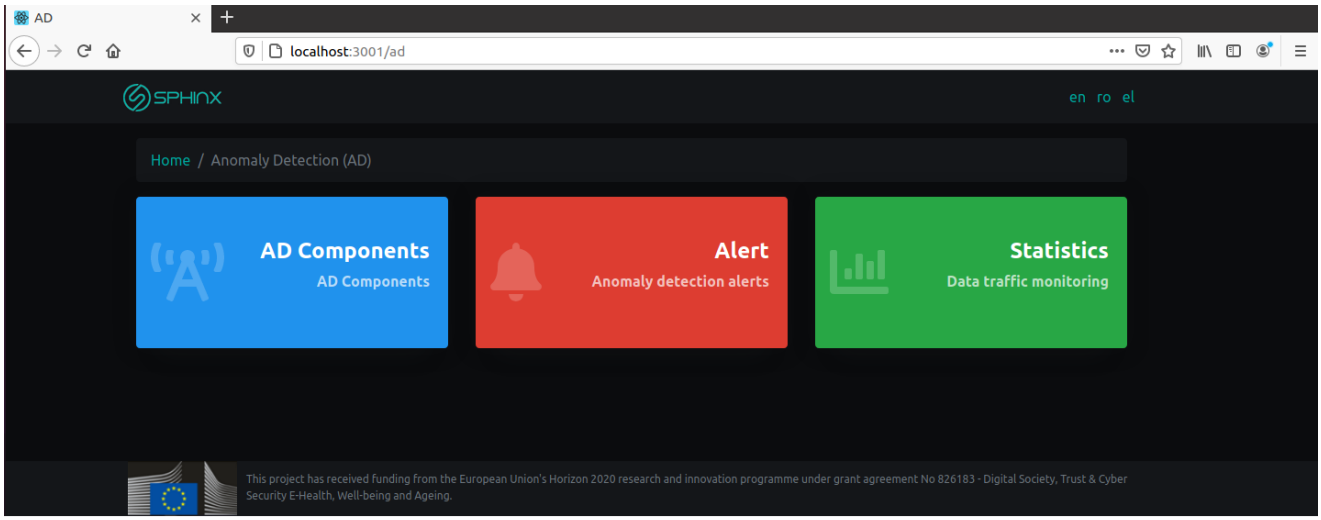



***Figure 4.15 SPHINX AD screen***

*AD Components* subcomponent contains the supported anomaly detection tasks. The user has the possibility to enable or disable them and to configure them. In the following image there is an example of detection tasks. This list is not final, in the second iteration of the component this list can suffer modifications.

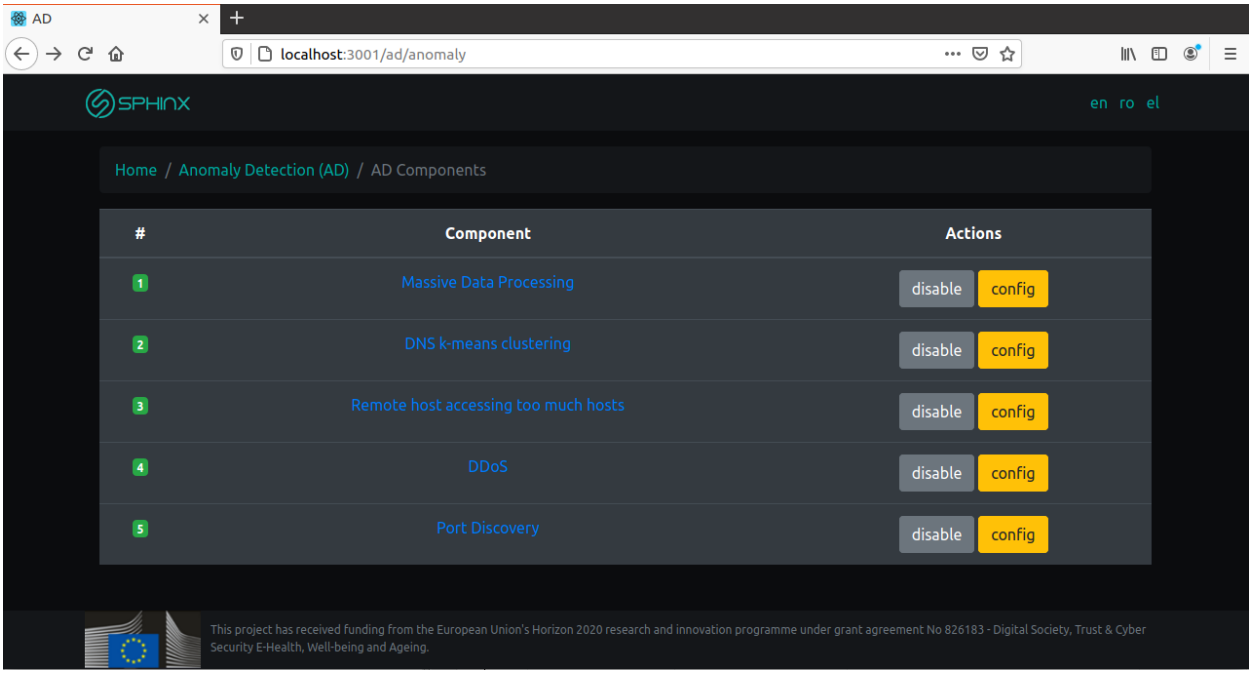



***Figure 4.16 SPHINX AD detection tasks***

For Massive Data Processing, the "config" button opens the configuration page where a certain threshold for transfer volume can be set and if it is exceeded, an alert is generated. This can be seen in the following figure.

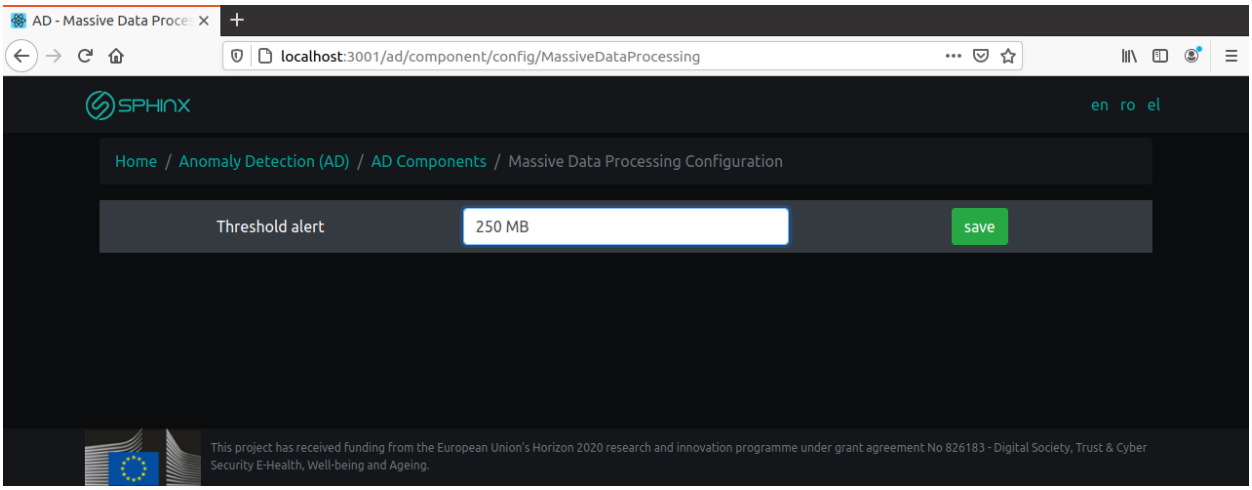


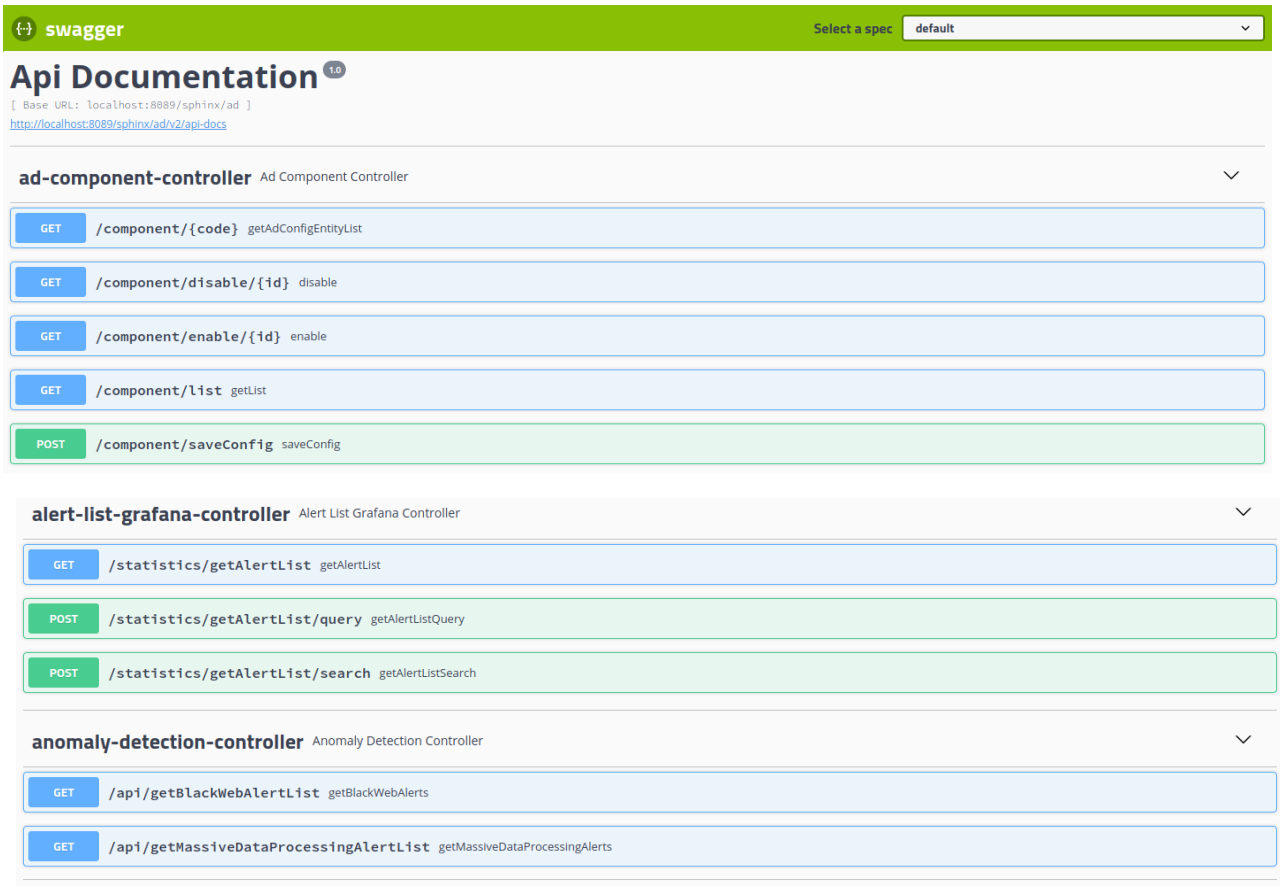

***Figure 4.17 SPHINX AD configuration page***

### 4.4.6 API

AD component uses REST services both for internal usage and for integration with external components.

Internally, the web front-end reads and writes data to the java backend.

The following figures present the API used by AD as generated by swagger.



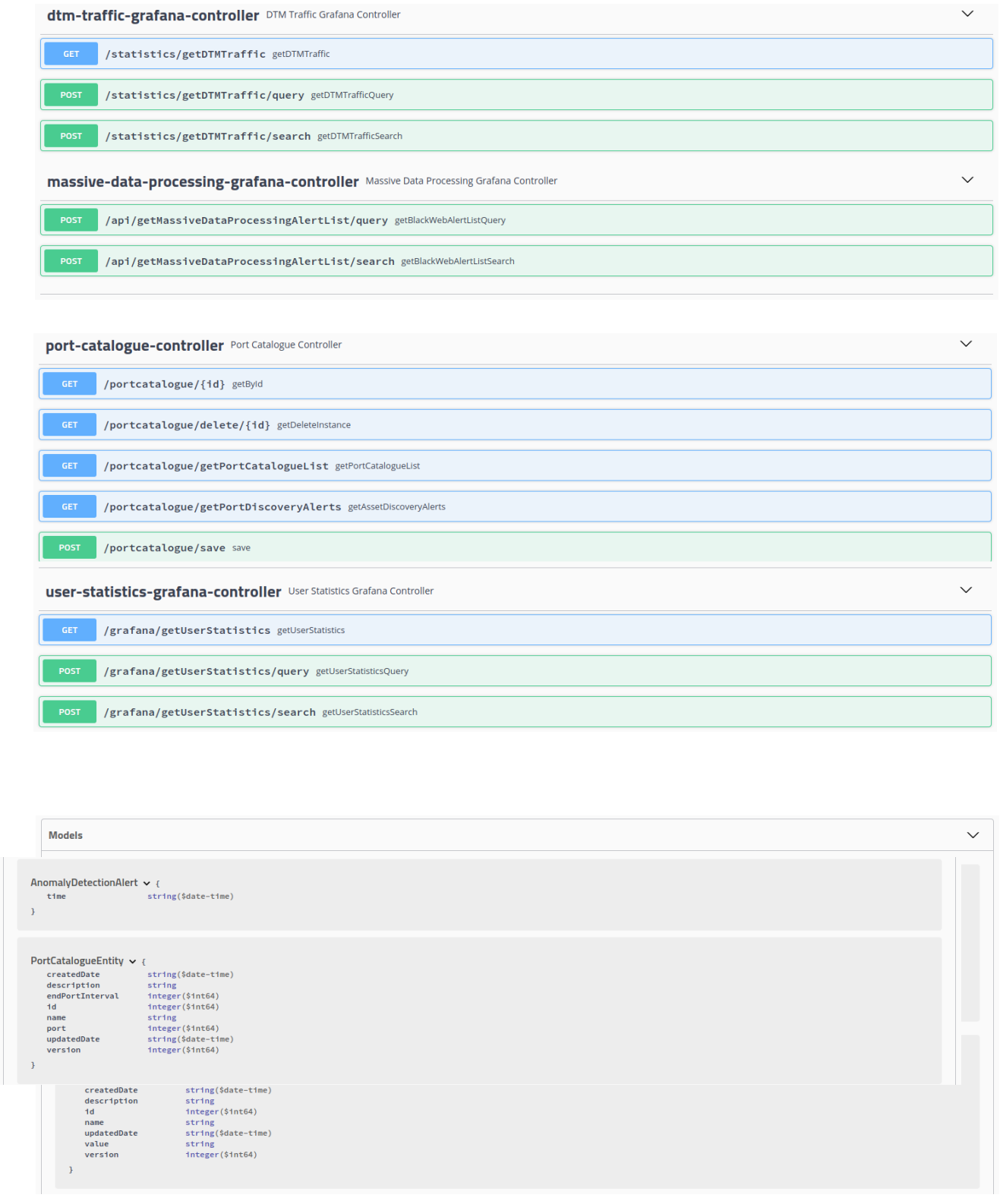



***Figure 4.18 SPHINX AD API***

## 4.5 Background

Anomaly-based intrusion detection systems were primarily introduced to detect unknown attacks, in part due to the rapid development of malware. The basic approach is to use machine learning to create a model of trustworthy activity, and then compare new behaviour against this model. Since these models can be trained according to the applications and hardware configurations, machine learning based method has a better generalized property in comparison to traditional signature-based IDS. Although this approach enables the detection of previously unknown attacks, it may suffer from false positives: previously unknown legitimate activity may also be classified as malicious. Most of the existing IDSs suffer from the time-consuming during detection process that degrades the performance of IDSs. Efficient feature selection algorithm makes the classification process used in detection more reliable.

Network-based anomalous intrusion detection systems often provide a second line of defence to detect anomalous traffic at the physical and network layers after it has passed through a firewall or other security appliance on the border of a network. Host-based anomalous intrusion detection systems are one of the last layers of defence and reside on computer end points. They allow for fine-tuned, granular protection of end points at the application level.

## 4.6 Anomaly Detection in Sphinx

AD component complements DTM in the role of detecting threat detection. Therefore, it's interaction with the other SPHINX components is similar.

AD supports the following interactions with the other components (as seen in figure 4.21):

- AD sends information on anomalies detected in system events and user behaviour that comprise a threat to the IT infrastructure to the following components: Forensic Data Collection Engine (FDCE), Security Information and Event Management (SIEM), Real-time Cyber Risk Assessment (RCRA), Knowledge Base (KB) and the Interactive Dashboards (ID).

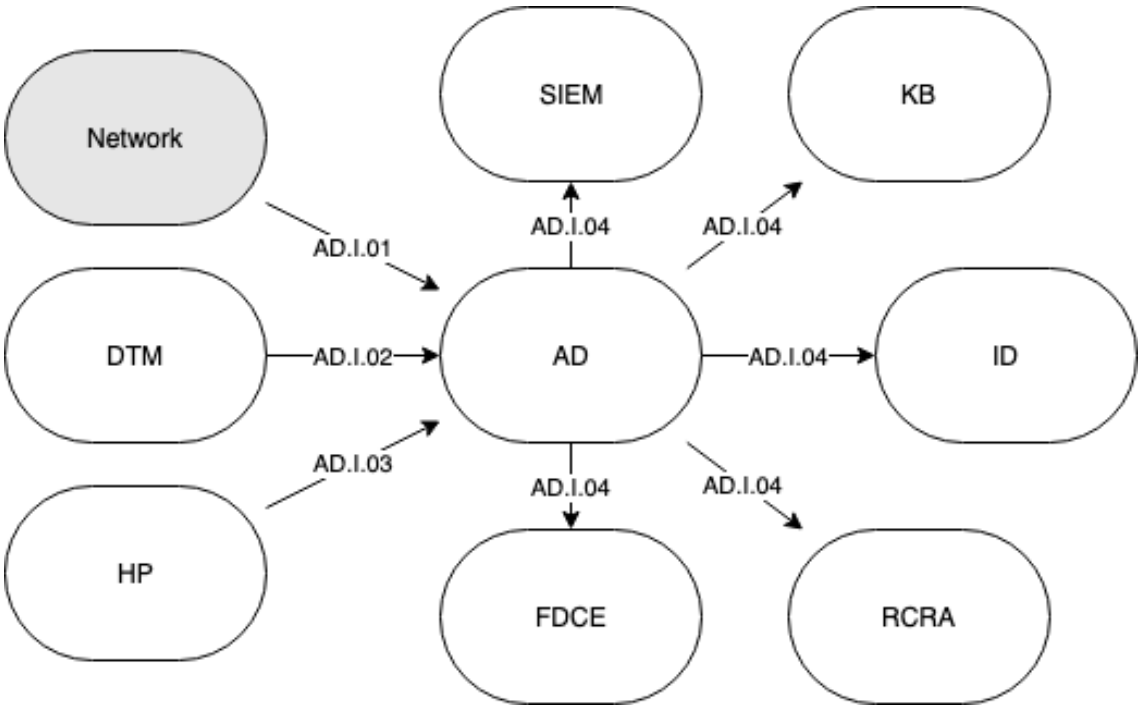



***Figure 4.19 SPHINX AD Collaboration Diagram***

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

### 4.6.1 Data collection

Data collection is done using the Data Traffic Monitoring component, which collects the network traffic from several protocols. The data that AD receives is a higher-level representation of the network activity. DTM + Suricata generate events, based on the raw traffic data, for http connections, DNS requests, FTP commands etc. The events contain relevant information for the type event so, for different types of events, the event contains different data.

For example, for http events, the information received by AD looks like this:


```
{
        "timestamp":"2020-07-02T12:11:15.362259+0300",
        "flow_id":831542229306266,
        "in_iface":"enp0s3",
        "event_type":"http",
        "src_ip":"10.0.2.15",
        "src_port":45976,
        "dest_ip":"172.0.19.99",
        "dest_port":80,
        "proto":"TCP",
        "tx_id":2,
        "http": {
            "hostname":"ocsp.pki.goog",
            "url":"\/gts1o1core",
            "http_user_agent":"Mozilla\/5.0 (X11; Ubuntu; Linux
x86_64; rv:79.0) Gecko\/20100101 Firefox\/79.0",
            "http_content_type":"application\/ocsp-response",
            "http_method":"POST",
            "protocol":"HTTP\/1.1",
            "status":200,
            "length":472
        }
}
```


### 4.6.2 Data aggregation and analysis

In order for ML algorithm to have good results, the data must be prepared. The features values should be normalized because some algorithms calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature.

Some anomaly detection tasks, like increased traffic usage, need data aggregated by time intervals. The spikes of traffic usage represent the anomalies and trigger alerts in AD components.

The main ML analyses tools are clustering algorithms, like k-means. The ML library in Apache Spark has strong support for a wide variety of ML algorithm, including clusterisation algorithms.

### 4.6.3 Visualization, alerting and real-time information

AD publishes relevant statistics and the alerts to Kafka. It also exposes the data through web-services.

Data visualization is based on Interactive Dashboards Component. Relevant graphs will be available in ID so the users can build dashboards with information that can help prevent and analyse security issues.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

# 5 Summary and Conclusions

This deliverable presented the current status of research work and the development direction for two SPHINX components: *Data Traffic Monitoring* and *Anomaly Detection*. It was explained that the two components are complementary, DTM detects known threats based on signatures, and AD is designed to detect unknown threats by creating profiles of normal behaviour and searching for network traffic events that are considered anomalous because they do not fit the profiles. The tools which are used in order to build the two SPHINX components were described and their advantages are explained.

This deliverable covers the first iteration of development for DTM and AD. The work on DTM and AD will continue during the second and final iteration. Lessons learned during the pilots and integration testing will be included in the development effort for the 2nd iteration.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.

# 6 References

[1] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, Kuang-Yuan Tung. *Intrusion detection system: A comprehensive review*

[2] Tshark - The Wireshark Network Analyzer 3.2.6 https://www.wireshark.org/docs/man-pages/Tshark.html

[3] Suricata | Open Source IDS / IPS / NSM engine https://suricata-ids.org

[4] Apache Hadoop https://hadoop.apache.org

[5] Apache HBase https://en.wikipedia.org/wiki/Apache_HBase

[6] Apache Spark https://en.wikipedia.org/wiki/Apache_Spark

[7] MLlib: Main Guide - Spark 3.0.0 Documentation https://spark.apache.org/docs/latest/ml-guide.html

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826183 - Digital Society, Trust & Cyber Security E-Health, Well-being and Ageing.