Analytic Engine User Manual

SPHINX

A Universal Cyber Security Toolkit for Health-Care Industry

Table of 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.

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.

1 Introduction

The AE is an essential part of the SPHINX Toolkit. Its purpose is to aggregate the data retrieved from other SPHINX components to provide helpful information to the end-users. The user will have the ability to interact with the ID and visualize the data through charts (e.g., pie, bar) for a predefined from him/her time interval. The information will be about how many events occurred, the event type, the most frequently attacked assets, etc., for the requested time interval. AE can also distribute its information to other SPHINX components through a powerful REST API.

2 Installation/Deployment

2.1 Overview

The SPHINX Analytics Engine consists of two main components.

1. The **Analytics Engine Database** for persisting historical data.
2. The **Analytics Engine API** that sits in front of the Analytics Engine Database and aggregates the stored data to provide useful analytics.

In order to deploy the Analytics Engine component in any system the following requirements should be met:

- Docker / docker-compose / kubectl and a K8S environment (eg. minikube) and kubectl
- Access to the internet
- Access to KT's container registry in Intracom's GitLab server
- Access to KT's repository in Intracom's GitLab server
- Git

2.2 Deployment using Docker

The easiest way to deploy the Analytics Engine component with docker is to clone the KT repository and use docker-compose:

- **git clone https://sphinx-repo.intracom-telecom.com/sphinx-project/decision-support-system/sphinx_dss**
- **cd sphinx_dss**
- **docker-compose up ae**

After running the above commands, the Analytics Engine API can be accessed on **<HOST IP>:4000.**

2.3 Deployment using Kubernetes

The SPHINX Analytics Engine component can also be deployed in a Kubernetes cluster by applying the deployments in the **kubernetes/local** directory. The **kubernetes/local** directory contains a simplified version of the actual deployments that can be deployed with zero configuration.

2.3.1 Start a Kubernetes cluster

For the sake of simplicity, we are going to use **minikube** in this guide to quickly start a local single-node Kubernetes cluster. Feel free to skip this part if you already have a Kubernetes cluster running.

To start the Kubernetes cluster simply run:

- **minikube start**

This will automatically configure the **kubectl** command-line tool to use the "minikube" cluster.

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.

2.3.2 Create a Secret with the credentials for the container registry

First, create a Secret called **intracom-repository** with your credentials for the container registry. This will allow the Kubernetes cluster to authenticate with the container registry and pull the docker images. You can create the secret by running the following command:

- **kubectl create secret docker-registry intracom-repository --docker-server=registry.sphinx-repo.intracom-telecom.com --docker-username=<your_username> --docker-password=<your_password> --docker-email=<your_email>**

Note: Replace **<your_username>** with your GitLab username, **<your_password>** with your GitLab password and **<your_email>** with your GitLab email address.

2.3.3 Start the Analytics Engine Database and the Analytics Engine API

Then, the Analytics Engine components can be started by executing the following commands:

- **git clone https://sphinx-repo.intracom-telecom.com/sphinx-project/decision-support-system/sphinx_dss**
- **cd sphinx_dss/kubernetes/local**
- **kubectl apply -f ae.yml**

2.3.4 Access the Analytics Engine API service

You can now access the component on **<HOST IP>:4000** by running:

- **kubectl port-forward -d <pod_name> 4000:5000**

Alternatively, you can run:

- **minikube service sphinx-ae-service**

and access the component on the host port that will automatically be assigned to the sphinx-ae-service.

3 Basic Case Example

For this tutorial we have two case examples for the SPHINX Analytics Engine usage. These case examples should help familiarize the reader with the SPHINX Analytics Engine's API. The user can use the "SPHINX KT" postman collection in order to go through the described cases and familiarize themselves with the API.

3.1 Setup

3.1.1 Login Authentication

First, the user should get a valid token from the Service Manager to authenticate their requests to the Analytics Engine API. This can be achieved by opening the "**Login**" request of the SPHINX KT postman collection and pressing "Send". This will automatically authenticate all the requests described 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.

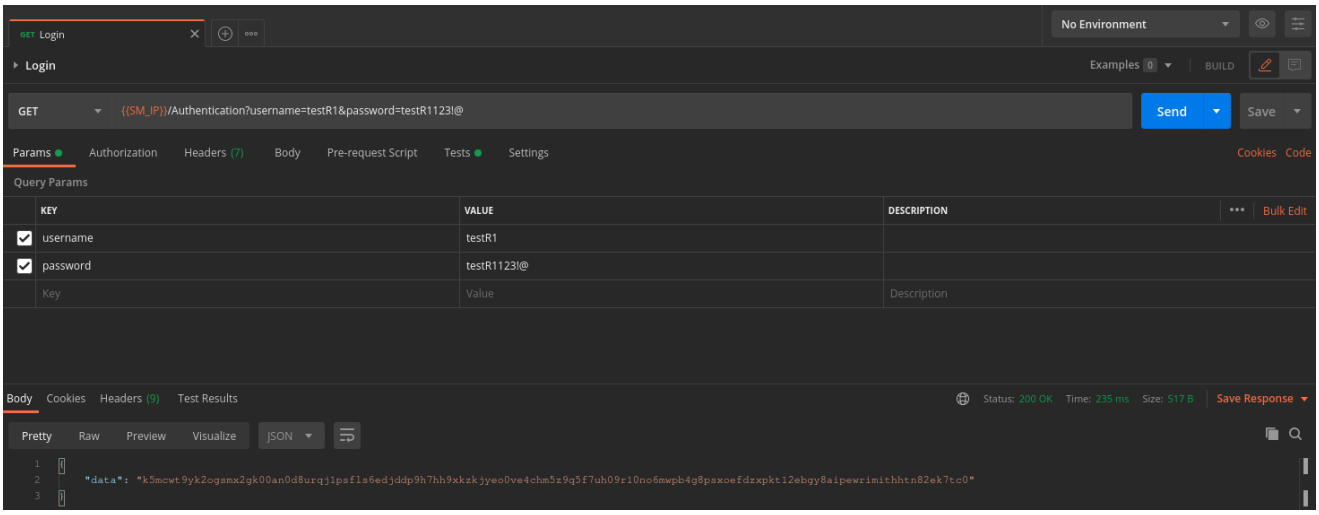

Figure 3.1 Login Request

3.1.2 Honeypot Data

Normally the HP component would POST data to the **/honeypots** endpoint for the attacks identified by the MLID component in the honeypot environment.

In case the HP component has not been deployed, the user can use the "**AE/HP - AE**" request in the SPHINX KT postman collection to simulate incoming HP traffic and populate the Analytics Engine Database with data.

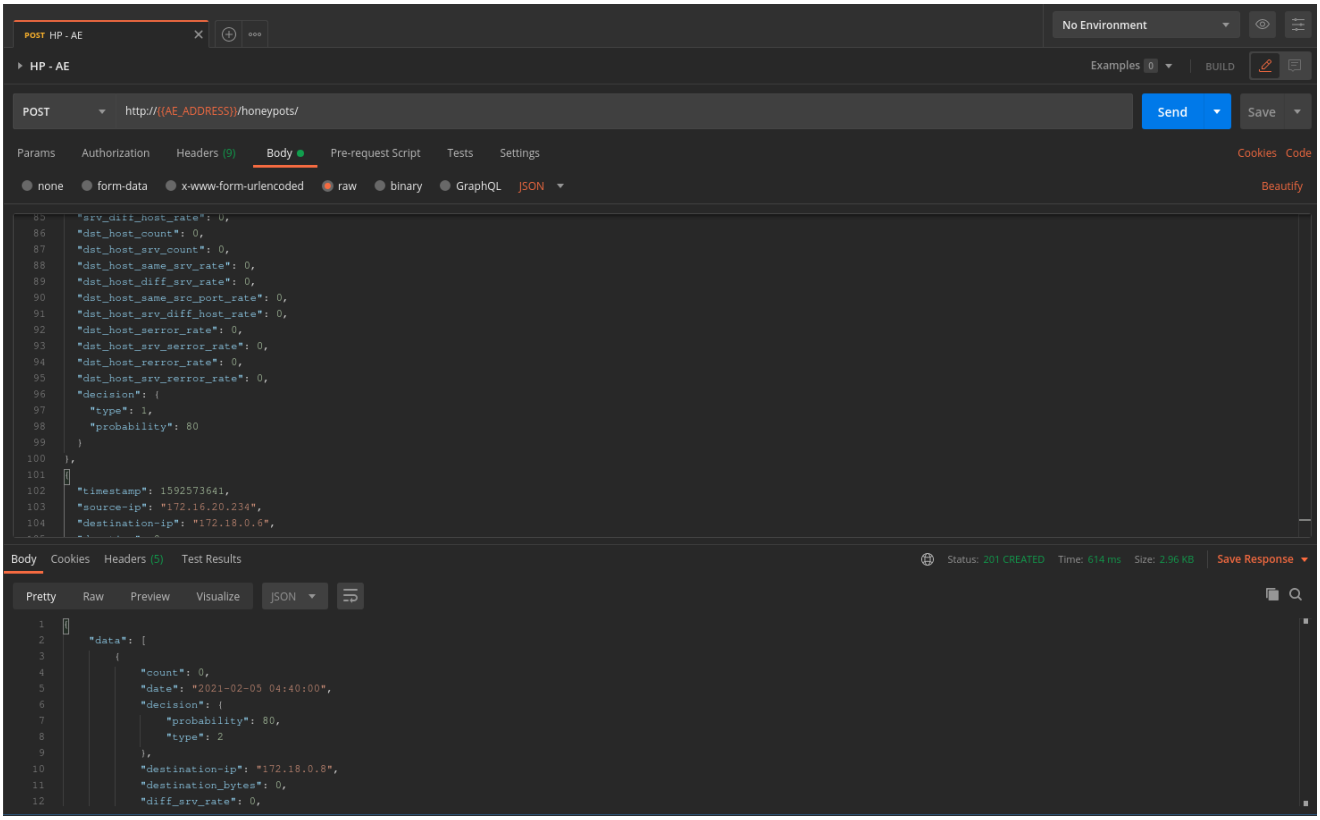

Figure 3.2 Data of Honeypot Endpoint

3.1.3 SIEM Data

Normally the SIEM would publish alerts to the **siem-alerts** Kafka topic.

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.

In case the SIEM component has not been deployed, the user can use the "**AE/Test/SIEM**" endpoint in the SPHINX KT postman collection to simulate incoming SIEM traffic and populate the Analytics Engine Database with data.

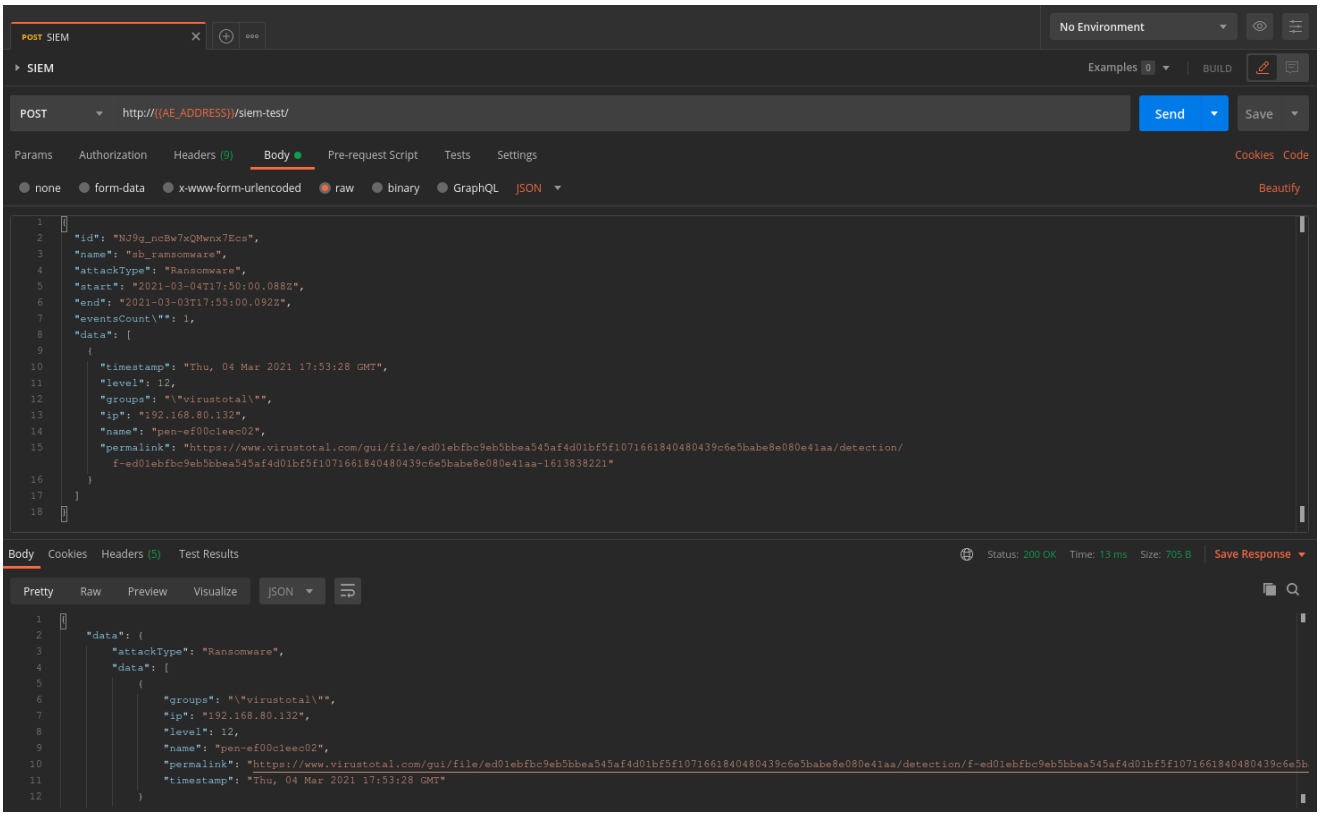

Figure 3.3 Data of SIEM Endpoint

3.1.4 DSS Data

Normally the DSS would publish alerts to the **dss-suggestions** Kafka topic.

In case the DSS component has not been deployed, the user can use the "**AE/Test/DSS**" endpoint in the SPHINX KT postman collection to simulate incoming SIEM traffic and populate the Analytics Engine Database with data.

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.

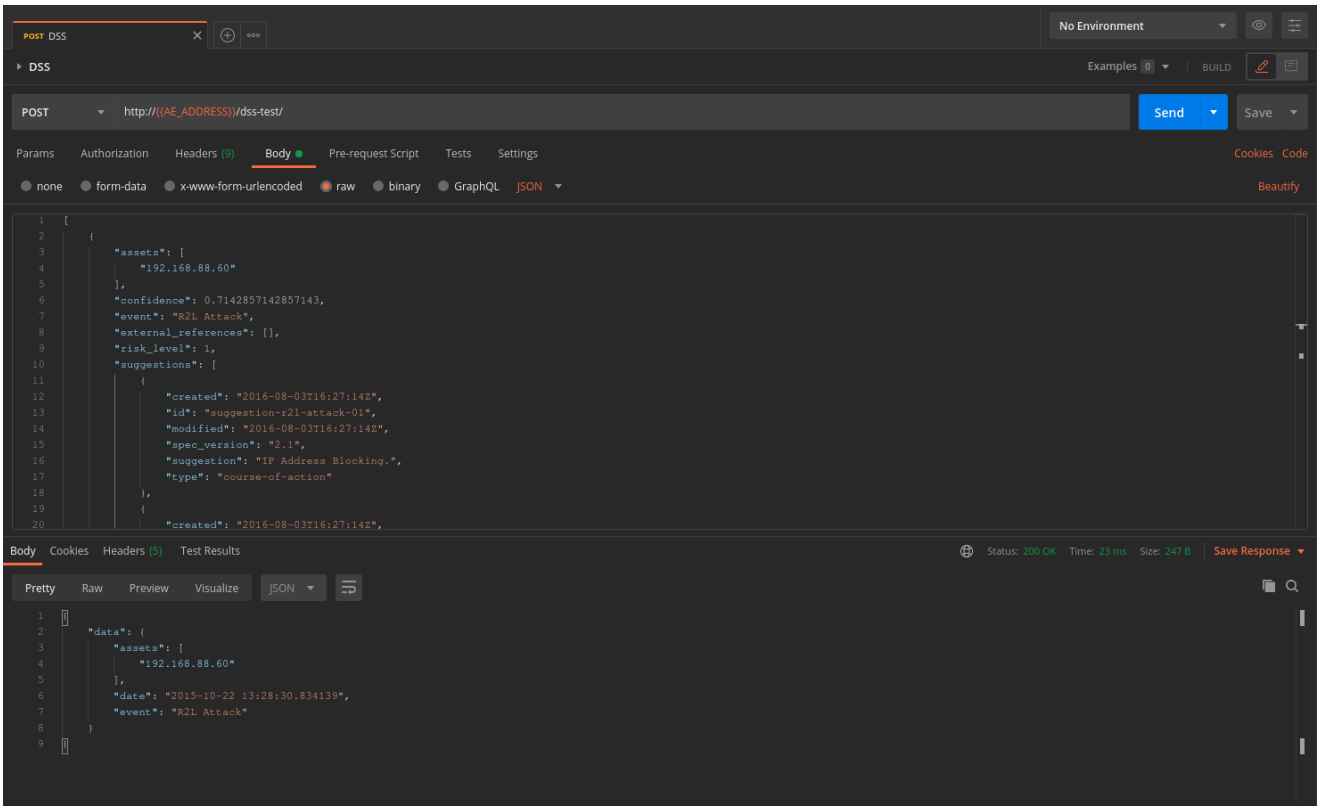

Figure 3.4 Data of DSS Endpoint

3.2 Case Example 1

3.2.1 Actor of the Case

The actor is either the Security Expert/IT of the hospital or any component of the SPHINX Toolkit (typically the Interactive Dashboards) that needs to fetch **attack-related analytics for the two environments** (simulated honeypot environment / operational environment).

3.2.2 Instructions

The actor sends a GET request to the root endpoint of the Analytics Engine (**/**).The actor specifies either a **time interval** or the **start_date** & **end_date** parameters.

- **time interval** is the number of days that the actor wants to fetch analytics for.

For example, setting **time interval** to 7 would generate attack related analytics for the previous week.

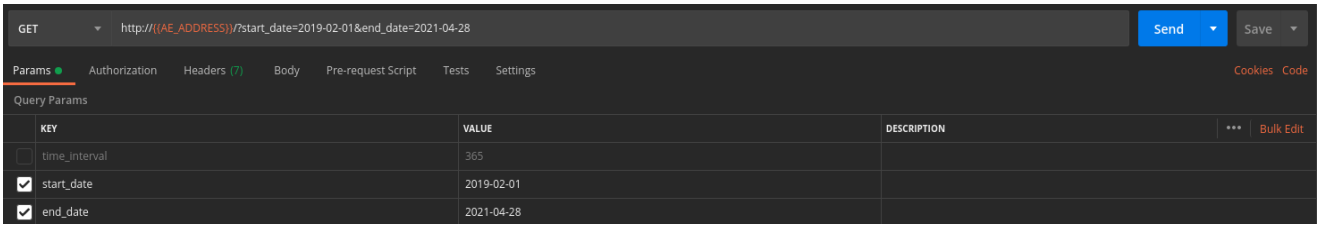

Figure 3.5 Analytics Based on start and end date

3.2.3 Expected Outcome

The Analytics Engine generates analytics for two separate environments, **hp** being the simulated honeypot environment and **operational environment** being the actual environment of the hospital.

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.

- **attacks** refer to the sum of attacks per attack type in the specified time range
- **attacks_per_day** refers to the sum of attacks performed on each date
- **attacks_per_ip** refers to the sum of attacks that have been performed against each specific hospital asset

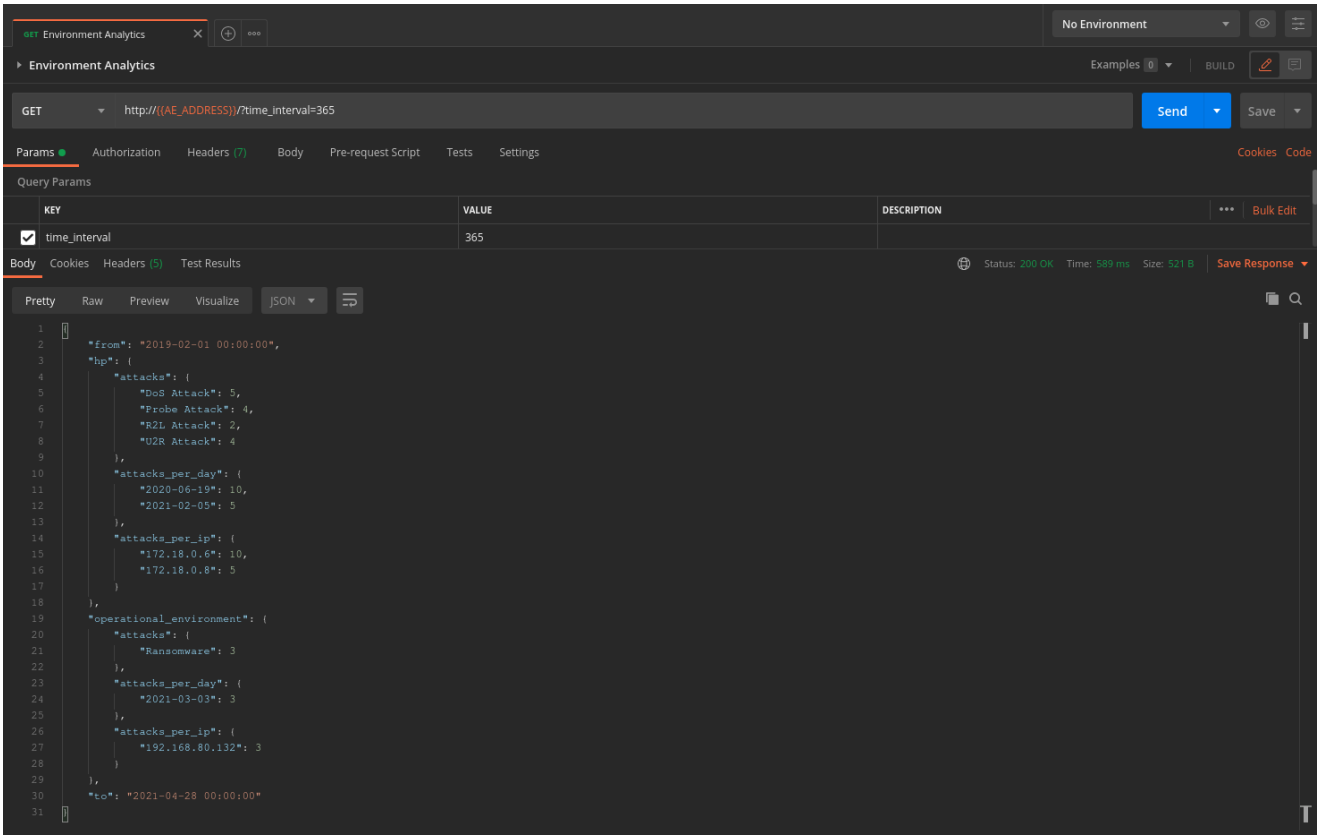

Figure 3.6 Outcome of Analytic Engine

3.3 Case Example 2

3.3.1 Actor of the Case

The actor is either the Security Expert/IT of the hospital or any component of the SPHINX Toolkit (typically the Interactive Dashboards) that needs to fetch **attack-related analytics for the output of individual tools** of the SPHINX Toolkit.

3.3.2 Instructions

The actor sends a GET request to an endpoint that refers to one of the SPHINX tools (**/honeypots/analytics**, **/siem/analytics**, **/dss/analytics**). The actor specifies either a **time_interval** or the **start_date** & **end_date** parameters just like in the previous case.

3.3.3 Expected Outcome

The Analytics Engine generates analytics for the output of the specified component. This way the actor can find out how many attacks were predicted by each one of the SPHINX tools.

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.

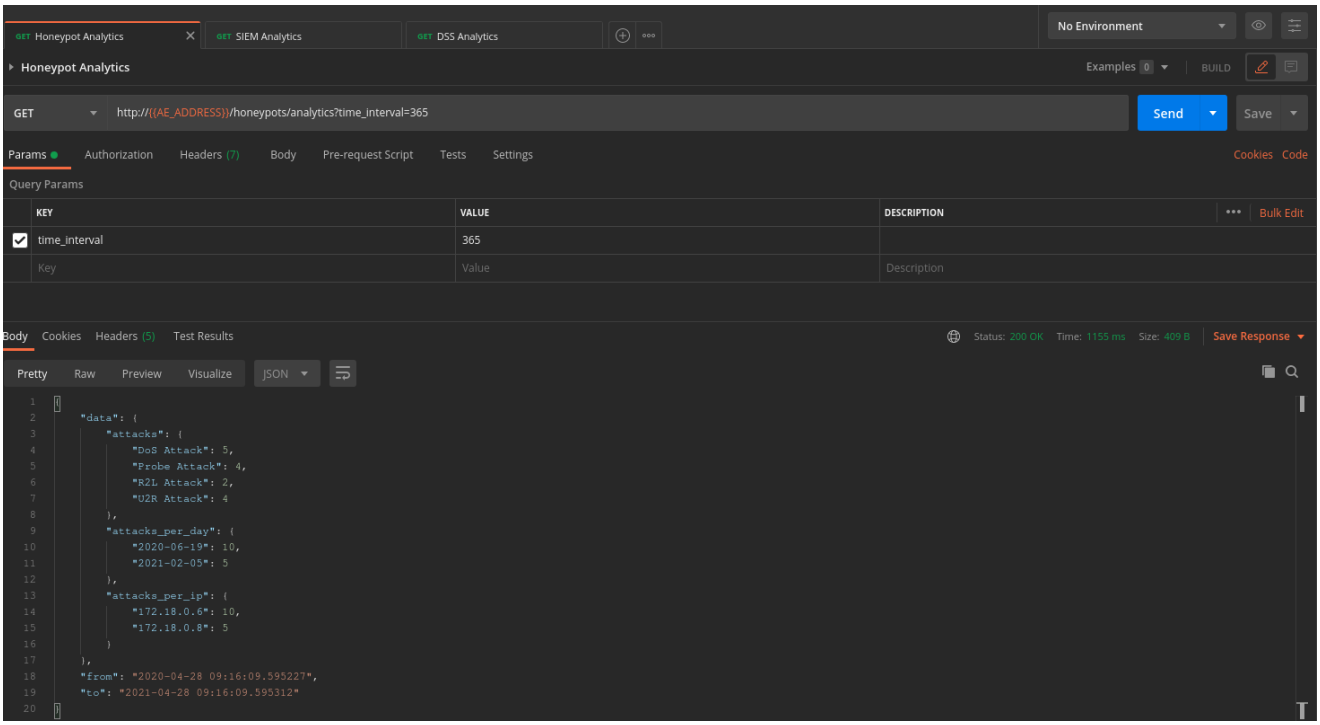

Figure 3.7 Case Example 2: Analytics based on Honeypot Data

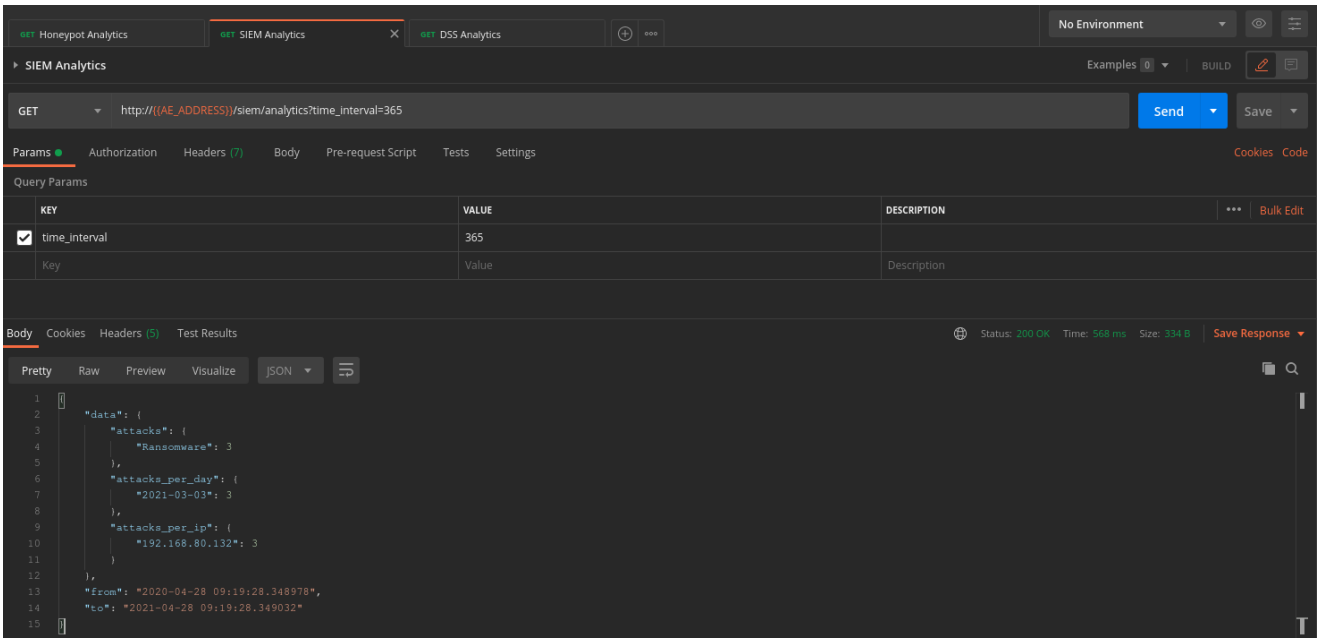

Figure 3.8 Case Example 2: Analytics Based on SIEM Data

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.

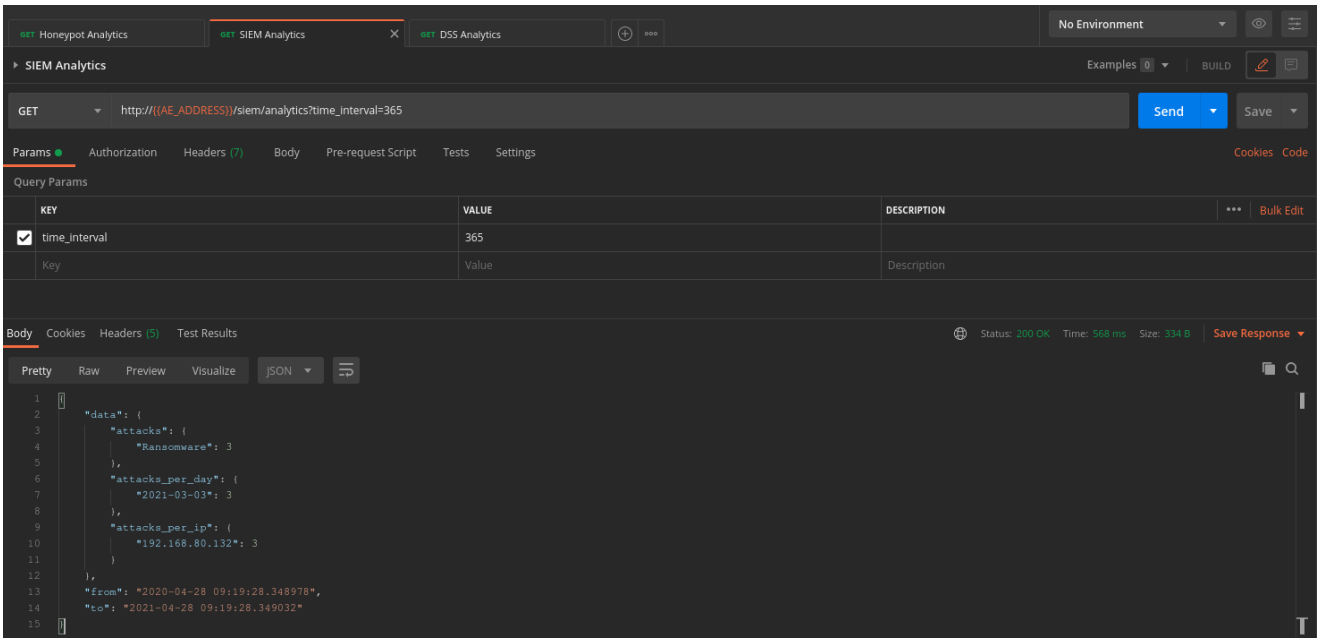

Figure 3.9 Case Example 2: Analytics based on DSS Data

3.4 Case Example 3

3.4.1 Actor for the Case

The actor is either the Security Expert/IT of the hospital or any component of the SPHINX Toolkit that needs to fetch the **raw output data of individual tools** of the SPHINX Toolkit.

3.4.2 Instructions

The actor sends a GET request to an endpoint that refers to one of the SPHINX tools (**/honeypots**, **/siem**, **/dss**). The actor specifies either a **time_interval** or the **start_date** & **end_date** parameters just like in the previous case.

3.4.3 Expected Outcome

The Analytics Engine API exposes the raw data that is stored in the Analytics Engine Database.

This way the actor can retrieve raw historical data for each of the 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.

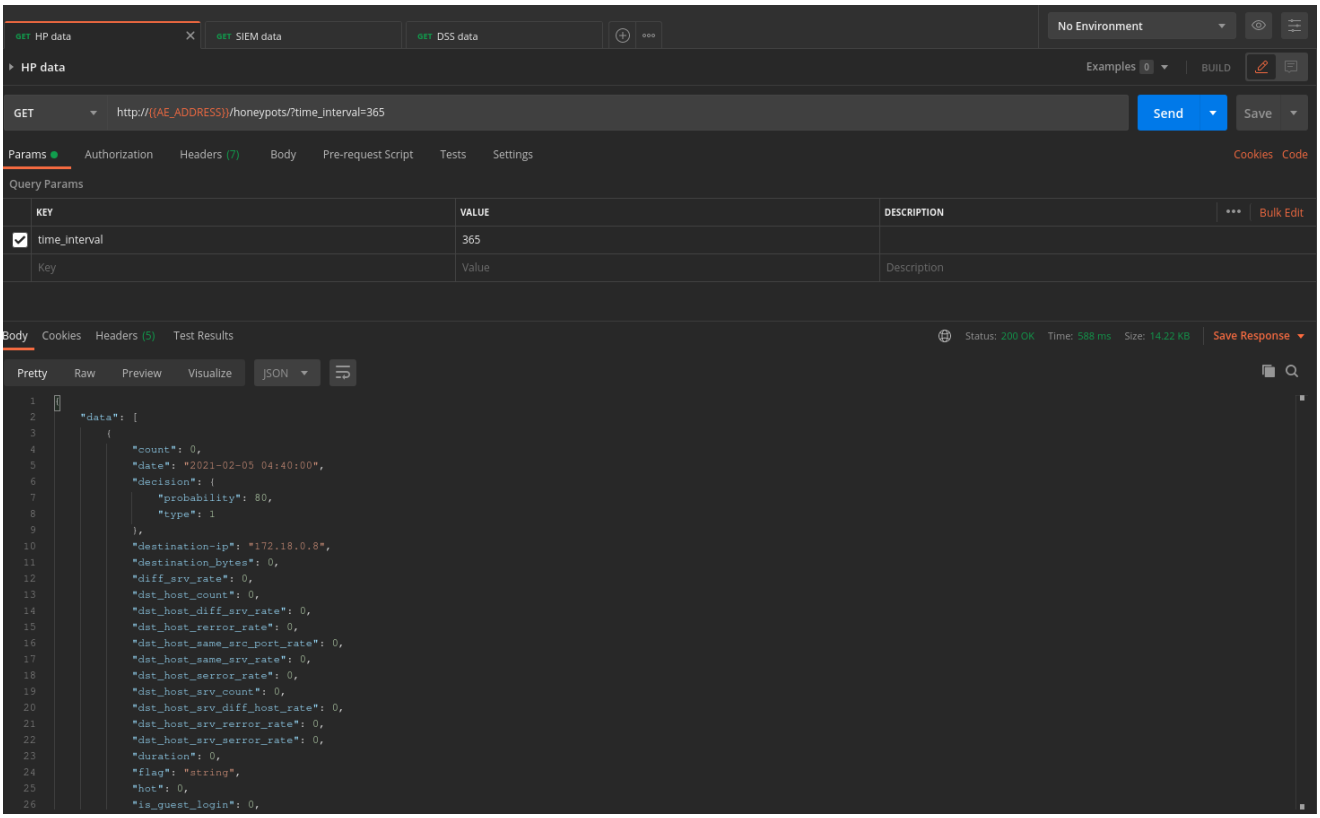

Figure 3.10 Case example 3: Analytics based on Honeypot Data

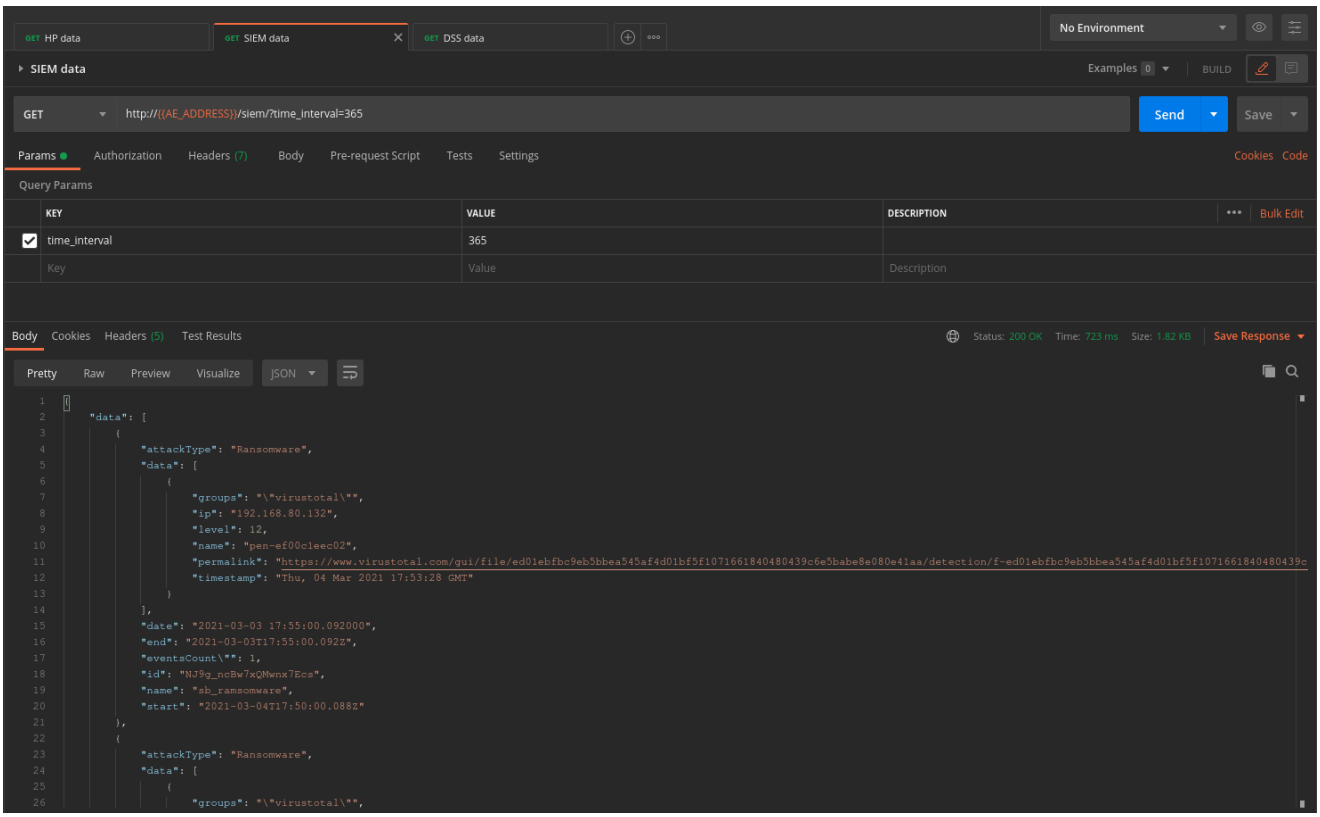

Figure 3.11 Case example 3: Analytics based on SIEM Data

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.

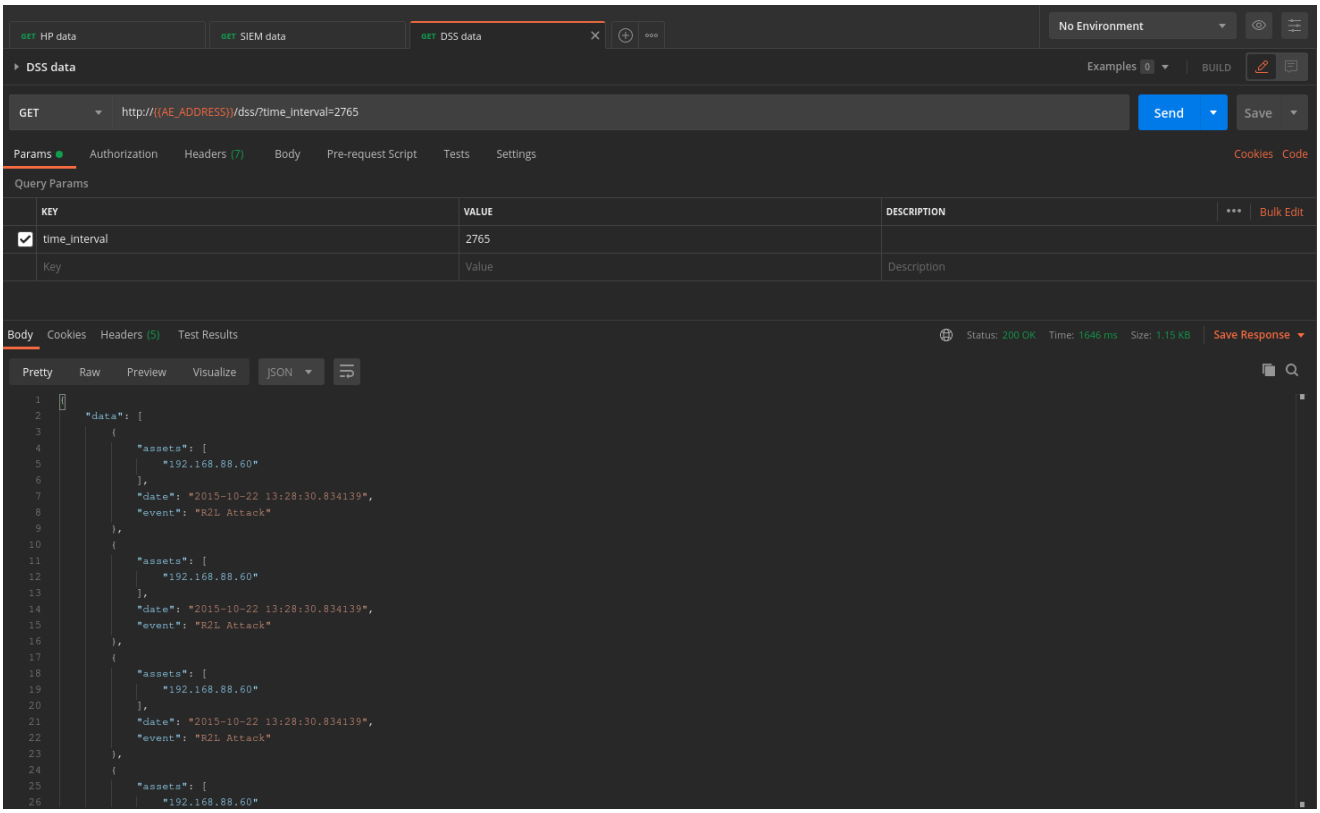

Figure 3.12 Case example 3: Analytics based on DSS Data

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.