UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

GENERACIÓ D'INTEL·LIGÈNCIA A PARTIR DE LA RECOPILACIÓ D'ATACS INFORMÀTICS

Autor: Jordi Bujaldón Devesa

Director: Dr. Raul Roca Canovas
Realitzat a: Departament Enginyeia Informàtica

Barcelona, 12 de juny de 2023

Abstract

In this project, I have developed a system to collect, analyze, and store data on real attacks captured from different honeypots. The work has been divided into three parts: the data source (honeypots), the database, and an integration program for these two platforms.

For the data source, I have used the T-Pot tool, which allows for the collection of a large volume of data of various types. Then, I installed the MISP platform, which is a specialized database for storing all kinds of incidents and threats. MISP was also installed alongside Cortex, which is a malware analyzer that detects malicious data. Once these tools were installed, I searched for a way to integrate them and make them work together. Therefore, I developed a Python program that connects to them through their respective REST API.

The integration program has been developed following good code design practices and applying different design patterns. Additionally, I have ensured to implement it in the most secure way to handle user credentials.

The combined work of these tools provides a solid platform for detecting, analyzing, and responding to threats, enabling more effective management of the collected information and the different tactics used by attackers.

Resum

En aquest projecte he desenvolupat tot un sistema per recopilar, analitzar i emmagatzemar dades sobre atacs reals capturats des de diferents honeypots. El treball l'he dividit en tres parts: la font de dades (honeypots), la base de dades, i un programa d'integració de les dos plataformes.

Per tal de poder desenvolupar el treball, una primera font de dades important ha estat una eina anomenada T-Pot que permet recopilar un gran volum de dades d'atacs de tot tipus. Després vaig instal·lar la plataforma MISP que tracta d'una base de dades especialitzada en emmagatzematge de tot tipus d'incidències i amenaces, a part d'altres funcions com detectar patrons i trobar correlacions entre amenaces; MISP també s'ha instal·lat conjuntament amb Cortex, que és un analitzador de malware que detecta si una dada és maliciosa. Un cop instal·lades aquestes eines, he buscat la manera amb la qual podia integrar-les i funcionessin de manera conjunta. És per això, que he implementat un programa amb Python que es connecta amb elles a través de les respectives API REST.

El programa d'integració l'he desenvolupat seguint unes bones pràctiques de disseny de codi aplicant diferents patrons de disseny. A més, he buscat la manera més segura de desenvolupar-lo per tal d'utilitzar les credencials d'usuaris.

El treball conjunt d'aquestes eines proporcionen una plataforma sòlida per detectar, analitzar i respondre davant d'amenaces, permetent una gestió més eficaç sobre la informació recopilada i les diferents tàctiques usades pels atacants.

Resumen

En este proyecto he desarrollado un sistema para recopilar, analizar y guardar datos sobre ataques reales capturados desde diferentes honeypots. El trabajo lo he dividido en tres partes: la fuente de datos (honeypots), la base de datos, y un programa de integración de las dos plataformas.

Para la fuente de datos he utilizado la herramienta T-Pot que permite recopilar un gran volumen de datos de todo tipo. Después instalé la plataforma MISP que es una base de datos especializada en almacenamiento de todo tipo de incidencias y amenazas. MISP también se ha instalado junto a Cortex, que es un analizador de malware que detecta si un dato es malicioso. Una vez estas herramientas han sido instaladas, he buscado la manera con la que podía integrarlas y que funcionasen de manera conjunta. Por eso, he desarrollado un programa en Python que se conecta a estas a través de sus respectivas API REST.

El programa de integración lo he desarrollado siguiendo buenas prácticas de diseño de código y aplicando diferentes patrones de diseño. Además, he buscado la forma más segura de implementarlo para utilizar las credenciales de los usuarios.

El trabajo conjunto de estas herramientas proporcionan una plataforma sólida para detectar, analizar y responder ante las amenazas, permitiendo una gestión más eficaz de la información recopilada y las diferentes tácticas usadas por los atacantes.

Agraïments

La realització d'aquest projecte no hauria estat possible sense l'ajuda de dues persones a les que voldria mencionar per donar-lis els meus agraïments.

Primer, al meu tutor Raul Roca que m'ha guiat durant tot el projecte i m'ha ensenyat un món totalment nou que m'ha agradat molt. Des d'un inici no em veia capaç d'afrontar el projecte, però amb la seva ajuda ha estat tot molt més fàcil.

I, en segon lloc, voldria agrair a la meva germana Anna que sempre s'ha preocupat per mi, m'ha aconsellat, i m'ha donat tot el suport poder tirar el projecte endavant.

Índex

1 Introducció

La gestió de la seguretat de la informació és cada cop més una prioritat tant per empreses com per organitzacions, i com a resultat al valor que proporcionen aquestes dades hi ha hagut un augment dels atacs cibernètics. Per això, és essencial prendre mesures efectives per protegir i garantir la integritat d'aquestes dades. Però, aquesta gestió de les dades no només es basa en protegir, sinó també en generar una resposta i una recuperació en cas d'haver un incident de seguretat.

Per aquest motiu existeix el que s'anomena *Intelligence-Driven Incident Response*, que consisteix en una estratègia per tota aquesta gestió de la informació que tracta de recopilar i analitzar les dades per, posteriorment, identificar, prevenir i respondre als atacs. La recopilació i utilització d'aquesta informació genera el que s'anomena *intel·ligència*, que gràcies a ella les organitzacions o empreses poden identificar patrons i tendències, tant en l'activitat de la xarxa, com en la securització de la informació per tal de prendre decisions més acurades, i respondre de manera més eficient i efectiva a les possibles amenaces de seguretat.

Aquest tema m'ha generat la curiositat per investigar i analitzar el seu funcionament, i a més realitzar un treball sobre la recopilació i anàlisi de dades sobre atacs per després generar intel·ligència.

Al començament no tenia cap coneixement sobre aquest tema: és per això que el meu tutor em va recomanar primerament llegir el llibre *Intelligence-Driven Incident Response* de Rebekah Brown i Scott J. Roberts, on vaig anar comprenent i agafant idees sobre el tema. Si ens centrem en el llibre ens parla sobre les diferents metodologies que s'utilitzen tant pels atacants per intentar complir les seves motivacions (ex. *Kill chain*), com pels defensors per intentar prevenir-les i, en el cas de ser atacats, dur a terme la recuperació dels sistemes de manera ràpida (ex. *F3EAD*).

En el meu cas, em vaig enfocar en la part dels defensors, i més exactament en la metodologia esmentada *F3EAD*. Aquesta metodologia consta de les etapes Find, Fix, Finish, Exploit, Analyze i Disseminate. Per aquest treball em vaig centrar en les quatre últimes etapes, ja que Fix i Finish són etapes que fan referència en un context d'un atac.

De manera resumida, les etapes en les quals he treballat consisteixen en:

1. Buscar dades útils que puguin proprocionar informació valiosa en un futur;
2. Recopilar-les, analitzar-les i integrar-les en sistemes de detecció i prevenció. d'amenaces;
3. Generar intel·ligència a partir de les dades;
4. Publicar aquesta intel·ligència per a que sigui d'ajuda per a altres organitzacions.

Al llarg del projecte em van sorgir diferents preguntes sobre si seria capaç de recopilar dades d'atacs reals, o analitzar les dades de manera que detectés qualsevol tipus d'amenaces. A mesura que he anat avançant el treball, mostraré com he anat responent a aquestes preguntes i quines solucions i implementacions he utilitzat per tal d'aconseguir-ho.

Un cop après tots aquests conceptes i metodologies, el meu tutor em va plantejar fer una base de dades d'atacs informàtics. En aquesta base de dades es guardarien les dades necessàries per a que, més endavant, es poguessin analitzar i detectar patrons que trobessin si algun sistema estava sent vulnerat.

2 Objectius

Després de tenir la idea principal del projecte es va definir uns objectius inicials que s'havien de dur a terme per tal de desenvolupar aquesta base de dades.
Per tal de guardar informació sobre atacs informàtics primer s'havia de trobar una font en la qual extreure les dades, el requeriment principal d'aquest objectiu era que aquesta font fos suficientment eficaç per poder recopilar el màxim de dades possibles.
En segon lloc, s'havia de trobar una base de dades que permetés emmagatzemar tot tipus de dades d'atacs.
Un cop obtinguda la base de dades i la font, s'havia de trobar la manera de guardar les dades recopilades de forma automàtica. Si es volia extreure el màxim volum de dades, aquest era un objectiu principal perquè fer-ho manualment no resultava gens viable degut a la quantitat d'informació que es recopilaria.

3 Planificació

Com s'ha vist anteriorment a l'apartat de la introducció, s'ha pogut intuir quines tasques s'ha anat desenvolupant al llarg del treball. En aquesta secció es mostrarà un llistat de les tasques que s'ha anat fent i els dies que s'han dedicat a cadascuna d'elles. Les tasques desenvolupades han estat:

- Investigació sobre el tema "Intelligence-Driven Incident Response"
- Instal·lació i investigació de les eines MISP i Cortex
- Instal·lació i investigació de T-POT
- Investigació sobre la API de Elasticsearch
- Investigació sobre llibreries de Python de Elasticsearch i MISP
- Implementació d'un programa d'integració

4 Generació d'intel·ligència i resposta incident

En el context actual de l'espai cibernètic és molt important estar preparats perquè en qualsevol moment podem ser atacats. És per això que quan hi ha qualsevol conflicte, és una prioritat generar una resposta ràpida i eficaç. Trobar la resposta adequada per cada cas és possible gràcies a la "Generació d'intel·ligència i resposta incident". En aquest apartat s'explicarà què és la intel·ligència, i què és la resposta incident, i es mostrarà com treballen de manera conjunta aplicant diferents mètodes per comprendre amb més profunditat les amenaces.

4.1 Introducció a la intel·ligència

La intel·ligència es basa principalment en agafar informació de fonts externes i analitzar-les seguint una sèrie de requeriments, per així posteriorment prendre una decisió. Abans, però, és important diferenciar dos conceptes com són dada i intel·ligència:

- Dada: Peça d'informació. És el que descriu alguna cosa. Un exemple de dada en l'àmbit informàtic podria ser una IP, domini, etc.
- Intel·ligència: Una dada per si sola no indica res, però quan vàries dades són recopilades i analitzades seguint uns requeriments, es converteixen en intel·ligència.

Sense anàlisi, moltes de les dades generades dins la indústria de la ciberseguretat continuaran sent dades. Però, quan aquestes són analitzades de manera correcta i es converteixen en intel·ligència, proporcionen un context necessari per respondre certes preguntes i ajuden a prendre decisions.

Hi ha varis mètodes d'on es poden extreure dades:

- Incidències i investigacions: Aquestes dades s'extreuen a partir d'activitats de respostes incidents fetes en passat i d'investigacions de violacions de dades. És un dels mètodes més utilitzats perquè els investigadors són capaços d'identificar aspectes de l'amenaça, així com eines i tècniques emprades a l'hora de la intrusió.
- Honeypots: Aquests sistemes estan configurats per emular màquines o xarxes senceres per recopilar informació a partir de la interacció amb aquests. La informació en els honeypots serà útil a mesura que es sapiga quina funció simula el sistema.
- Fòrums i pàgines web: Hi ha empreses que entren a la "deep web" o "dark web" per recopilar dades dels fòrums o sales de xat que hi ha. En aquests llocs es comparteix molta informació útil sobre atacs.

Per generar intel·ligència existeix un model de procés que s'anomena Çicle d'intel·ligència". Aquest cicle conté vàries etapes:

1. Direcció: Estableix la pregunta que l'intel·ligència vol respondre.
2. Recopilació: Aquesta etapa tracta de reunir tantes dades com sigui possible, i si pot ser, de diferents fonts. La informació redundant també aporta valor. Això ens porta a construir una gran capacitat de dades que inclou tant informació tàctica, com d'infrastructura, malware, vulnerabilitats, informació estratègica, etc. Aquesta etapa és un procés llarg i es repeteix constantment.
3. Processament: Les dades normalment no venen en el format que volem com en JSON, XML, CSV o, fins i tot, correus. Els tipus de processament de dades poden ser normalització, indexació, translació o filtració.
4. Anàlisi: Busca la resposta a la pregunta formulada a l'etapa de Direcció. El tipus de resposta dependrà de com s'hagi formulat aquesta pregunta. Pot ser una resposta en format d'informe, o un si/no, etc.
5. Difusió: Un cop s'ha obtingut la resposta amb el format correcte, aquesta serà útil quan arribi als destinataris adients. Aquests poden ser tant superiors dins d'una empresa com programes de detecció d'intrussos o firewalls.
6. Feedback: Aquesta etapa valida si la resposta generada és correcta o no. Si la resposta és exitosa normalment el cicle acabarà, però pot donar-se el cas que obri portes a noves preguntes i torni a començar un nou cicle. En cas que hagi fracassat, s'han d'identificar els errors i tornar a començar.

Quan es recopila la informació és important que aquesta s'entengui, com més detalls es tingui sobre la informació, millor serà el procés d'anàlisi. Una altra recomanació és adjuntar la data en la que es va trobar la dada, i establir-li un context. És important saber en quin context fa referència per poder entendre-la amb més profunditat.

4.2 Introducció a la resposta incident

La resposta incident abarca el procés d'identificar intrusos, desenvolupar informació necessària per entendre'ls i executar plans per eliminar-los. Igual que la intel·ligència, la resposta incident també té varis models d'actuació però tracterm el cicle anomenat F3EAD. El nom fa referència a les inicials Find (Buscar), Fix (Arreglar), Finish (Finalitzar), Exploit (Explotar), Analyze (Analitzar) i Disseminate (Difusió), i aquestes etapes funcionen de la següent manera:

- Find: Aquesta fase determina les amenaces a la que ens enfrontarem. Aquestes amenaces es poden trobar a través de diferents fonts. Es fa una cerca activa i la recopilació de dades rellevants sobre les amenaces que es volen tractar.

- Fix: Després d'obtenir la informació necessària, aquesta fase tracta de determinar en quin punt es troba l'amenaça. A partir d'aquí, s'ha de cercar dins de la xarxa, sistemes, serveis, etc, si hi ha algun compromès.

- Finish: Aquesta fase inclou la resposta incident. Es prenen decisions d'actuació sobre l'adversari intentant contenir-lo, mitigar-lo i erradicar-lo.

- Exploit: L'objectiu d'aquesta etapa és reunir el màxim d'informació sobre l'incident per poder generar intel·ligència a partir de l'atac. Alguns exemples de dades que es podrien recopilar són:

  - Indicadors de compromís (IP, URLs, hashes...)
  - Exemples de malware o vulnerabilitats atacades
  - Informes d'usuaris i de les incidències
  - Comunicacions de l'atacant
  - Objectius de l'atac, motivacions, etc.

- Analyze: A partir de les dades recopilades en la fase d'Exploit, es desenvolupa aquesta informació seguint diferents metodologies. Aquestes podrien ser "kill chains", anàlisi de malware, etc. La "kill chain" són una sèrie de passos que normalment els adversaris utilitzen per atacar els seus objectius. Aquesta fase pot portar a trobar més indicadors de compromís, que aquests a la vegada poden portar a trobar més dades de l'atac. El més important és trobar aquelles tècniques, tàctiques i procediments que ha utilitzat l'adversari contra nosaltres.

- Disseminate: Per últim, aquesta fase es basa en mostrar els resultats al públic. Pot ser una resposta més tàctica per a que es pugui dur a terme una resposta incident, també pot ser més estratègica per tal de prendre decisions en un futur o pot ser més orientada a tercers, ja que les organitzacions o empreses a vegades comparteixen grups d'intel·ligència.

5 Desenvolupament

Per a aquest projecte s'ha investigat i explorat una integració de dues eines que s'utilitzen per trobar, recopilar i analitzar amenaces o incidents de seguretat. Aquestes eines són T-Pot, MISP i Cortex: T-Pot és una plataforma que atrau i recopila dades sobre activitat maliciosa, i MISP és una plataforma d'intercanvi i compartició d'informació sobre amenaces que permet analitzar les dades amb l'ajuda de Cortex.

La integració de T-Pot i MISP permetrà capturar i analitzar la informació d'activitat maliciosa detectada a T-Pot, i compartir-la amb la comunitat a través de MISP. La idea principal és aprofitar les capacitats d'aquestes eines per millorar la detecció i l'anàlisi d'amenaces.

A continuació s'explicarà amb més detall com funcionen les eines esmentades anteriorment, com s'han configurat per a aquest projecte i quins són els resultats que proporcionen; després, es mostrarà la implementació del programa que les integra i s'explicarà les decisions que s'han pres a l'hora de dissenyar l'estructura del programa.

5.1 T-Pot

T-Pot és una plataforma que proporciona una implementació de diversos honeypots preconfigurats, on cadascun d'ells representen ser diferents tipus de sistemes i serveis. Està basat en Debian 11 (Bullseye) i utilitza diferents dockers per aconseguir l'execució dels diferents sistemes per a que s'executin de manera paral·lela.

5.1.1 Honeypots

En l'àmbit de la ciberseguretat, els honeypots són eines que s'utilitzen sobretot en els IDS (Intrusion Detection System). Es fan servir tant per monitoritzar com per analitzar activitats malicioses a les xarxes i el seu principal objectiu és atraure atacants i recopilar tota la informació possible per saber sobre les seves tàctiques i tècniques.

Un honeypot és una trampa que simula ser un sistema amb vulnerabilitats. Aquestes trampes normalment són configurades per atraure atacants, els quals intenten comprometre el sistema. La configuració d'aquests honeypots sempre es fa en un entorn aïllat i segur de l'entorn de treball principal d'una empresa, organització, etc, i permet als responsables de la ciberseguretat recopilar informació sobre els intents d'intrusió, comportaments i eines utilitzades per part dels atacants. T-Pot ofereix 22 honeypots diferents, a part d'altres eines que complementen els honeypots per obtenir la millor aplicació d'aquests. Els honeypots i eines amb els quals s'ha treballat en aquest projecte han estat els següents:

- Tanner: És un honeypot que simular ser una aplicació web i va complementat de Snare, que a la vegada és qui mana a Snare sobre com actuar davant les peticions que se li facin. Cada esdeveniment que rep Snare és evaluat per Tanner i ell és qui decideix la següent acció per dur a terme.

- Suricata: Aquesta eina és un motor de monitorització de xarxa que detecta activitats malicoses i possibles amenaces. Aquest motor es complementa molt bé amb els honeypots, ja que es pot veure l'activitat de la xarxa que hi ha dirigida en un honeypot en concret.

- Fatt: Script construït amb Python utilitzant la llibreria tshark que extreu les metadades de la xarxa com els JA3, HASH, capçaleres HTTP, etc. El principal ús que té aquest programa és la monitorització de honeypots, encara que també es pot utilitzar en l'anàlisis forense.

5.1.2 Arquitectura

La manera d'instal·lar aquesta plataforma de forma aïllada va ser creant una màquina virtual pujada a un servidor d'Azure. Dins d'aquesta màquina virtual es va instal·lar T-Pot que consta de la següent arquitectura:

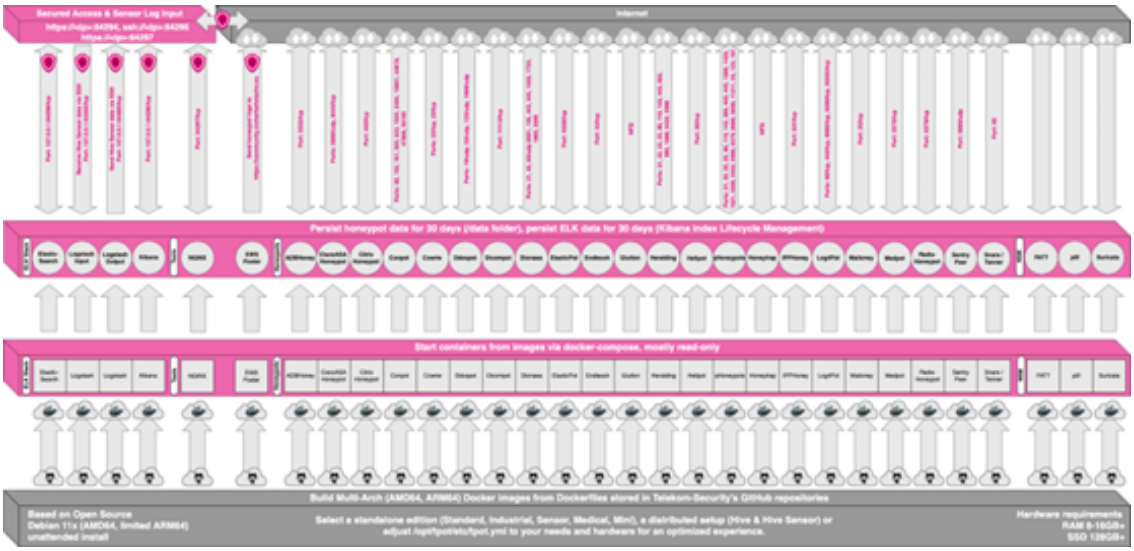

Figura 1: Arquitectura de la plataforma T-Pot

La imatge anterior mostra l'arquitectura interna de la plataforma. En la part superior es pot veure quines són les adreces i ports que s'utilitzen en cada un dels honeypots i serveis que ofereix.

Les principals adreces que s'han de tenir en compte són: https://20.26.120.58:64297 i https://20.26.120.58:646294, amb 20.26.120.58 com a IP pública de la màquina virtual. Aquestes dues adreces permeten a l'usuari interactuar amb la plataforma de forma còmoda i ràpida.

La primera adreça referenciada permet accedir a la plataforma a través d'una pàgina web obrint-li a l'usuari un portal amb diferents opcions per escollir sobre els serveis de T-Pot:

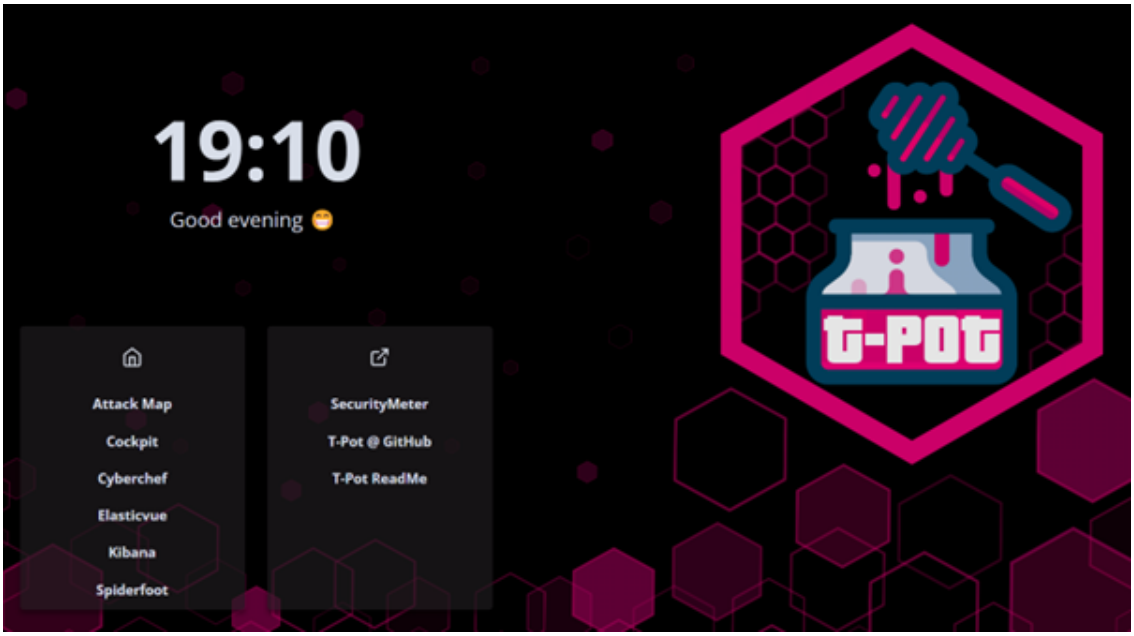

Figura 2: Pàgina principal de T-Pot

La pàgina principal permet a l'usuari navegar a diferents serveis. El primer, anomenat "Attack Map", mostra a l'usuari un mapa geogràfic on s'observen els atacs a temps real que s'estan fent en aquest moment sobre el honeypot. En la següent imatge es mostren dos atacs produïts amb les dades corresponents, el països des d'on s'ha atacat, IP de l'atacant, el honeypot al qual s'ha atacat, i el protocol utilitzat:

Figura 3: Mapa d'atacs a temps real

L'opció de "Cockpit" de la pàgina principal redirigeix a l'usuari a la segona adreça referenciada anteriorment, a https://20.26.120.58:646294. Aquesta adreça permet a l'usuari interactuar amb el sistema operatiu de la màquina virtual (que és Debian 11 com s'ha dit anteriorment):

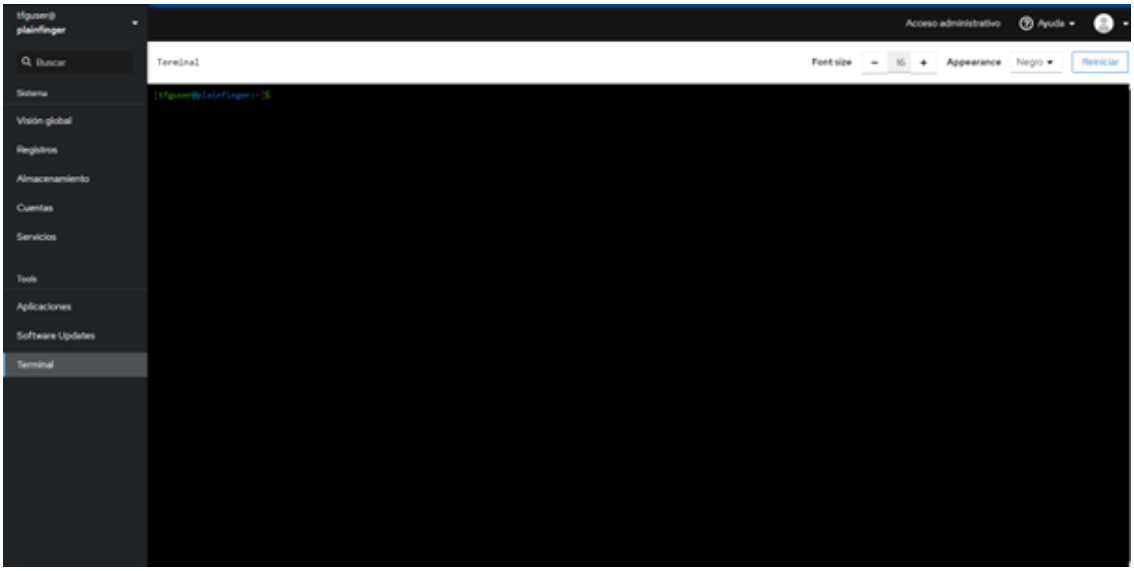

Figura 4: Consola de la màquina virtual

L'avantatge de tenir aquesta opció és la facilitat amb la que et permet connectar a la consola de la màquina virtual i poder accedir als seus recursos, a part de poder configurar la plataforma en el cas que sigui necessari. A més, aquesta pàgina proporciona un altre tipus d'informació com és la capacitat de recursos de la màquina o processos que s'estan executant.

La secció de "Cyberchef" no s'ha utilitzat en aquest projecte, però és una aplicació que permet encriptar, codificar, comprimir i analitzar dades.

L'opció d'"Elasticvue" és una pàgina web que permet cercar i interactuar amb el cluster d'Elasticsearch que té T-Pot integrat. Dins de l'Elasticsearch és on es guarda tota la informació recopilada dels atacs. Per buscar la informació, s'utilitza l'API de cerca d'Elasticsearch. Per exemple, si volem buscar els primers 1000 esdeveniments que s'han produït a Suricata, hem d'executar la següent comanda:

```
{
    "query": {
        "query_string": {
            "query": "Suricata"
        }
    },
    "size": 1000,
}
```

Amb aquesta comanda retorna tots els "hits" que troba dins d'Elasticsearch que continguin la paraula "Suricata". D'aquesta forma és com el programa d'integració que s'ha implementat obté les dades de T-Pot. Els hits retornats és una llista d'elements JSON que contenen les dades dels esdeveniments. Un exemple d'un JSON de Suricata podria ser el següent:


```
"_source": {
    "geoip": {
        "country_code3": "US",
        "country_name": "United States",
        "continent_code": "NA",
        "timezone": "America/Chicago",
        "ip": "20.97.164.253",
        "latitude": 37.751,
        "longitude": -97.822,
        "location": {
            "lat": 37.751,
            "lon": -97.822
         },
         "country_code2": "US"
    },
    "t-pot_hostname": "plainfinger",
    "type": "Suricata",
    "event_type": "ssh",
    "in_iface": "eth0",
    "proto": "TCP",
    "src_port": 53964,
    "ssh": {
        "server": {
            "software_version": "OpenSSH_7.9p1",
            "hassh": {
                "hash": "b74b3746d7c1b9944b2e8db18f062e6f",
                "string": "curve25519-sha256,curve25519-sha256@libssh.org..."
              }
        },
        "client": {
            "hassh": {
                "hash": "4e066189c3bbeec38c99b1855113733a",
                "string": "curve25519-sha256,curve25519-sha256@libssh.org..."
            }
        }
    },
    "st_port": 22,
    "ip_timestamp": "2023-05-05T22:27:31.220427+0000",
    "de_rep": "known attacker",
}
```

Aquest és un JSON amb alguns dels camps que pot tenir els elements de Suricata. Depenent del honeypot o eina que estigui sent atacat, els camps variaran amb la informació rellevant d'aquell sistema. En aquest cas, veiem alguna informació com la informació geogràfica de l'atacant com la seva IP, el país, la localització en coordenades, etc. Si per exemple, agafem aquesta IP i la comprovem en pàgines com VirusTotal.com, es veu perfectament que hi ha diversos analitzadors de malware que identifiquen aquesta IP com a maliciosa:

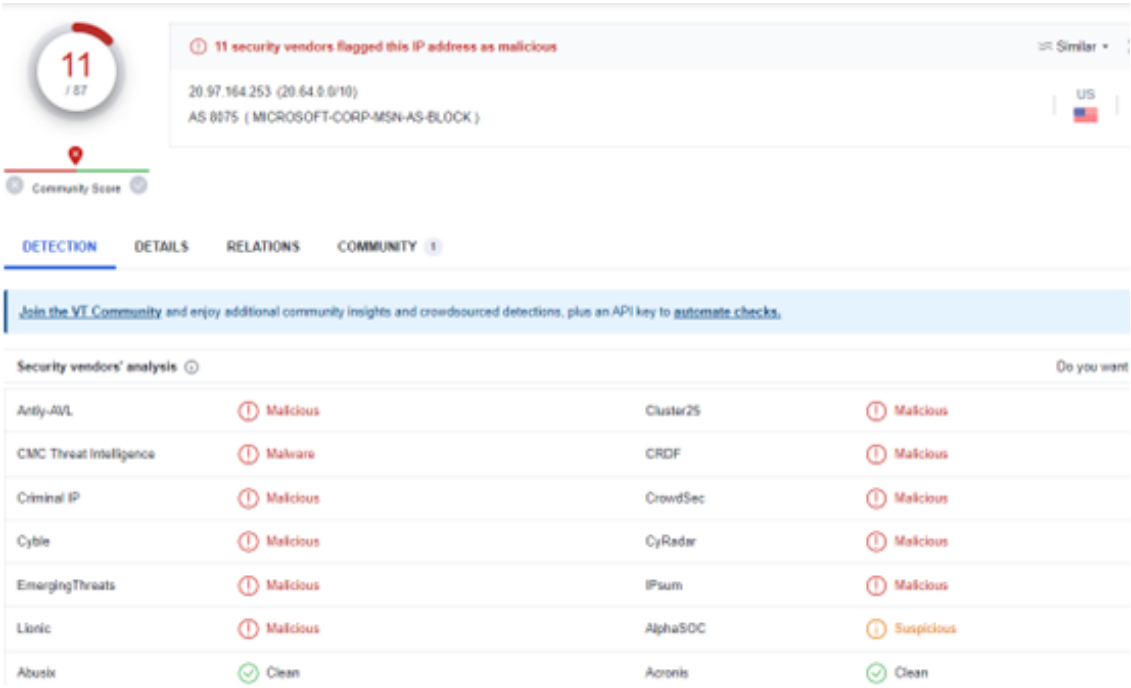

Figura 5: Detecció d'amenaces amb VirusTotal

Tots els honeypots comparteixen alguns camps: un d'aquests és el camp "type", que indica a quin honeypot fa referència l'esdeveniment, per tant, és amb aquest camp on trobem els esdeveniment referents als honeypots amb la crida de l'API d'Elasticsearch.

A part de totes aquestes eines que proporciona T-Pot, també té un visualitzador de dades d'Elasticsearch amb el programa Kibana; i gràcies a aquesta opció es poden veure de manera ordenada i detallada la informació dels atacs produïts per cada honeypot o eina:

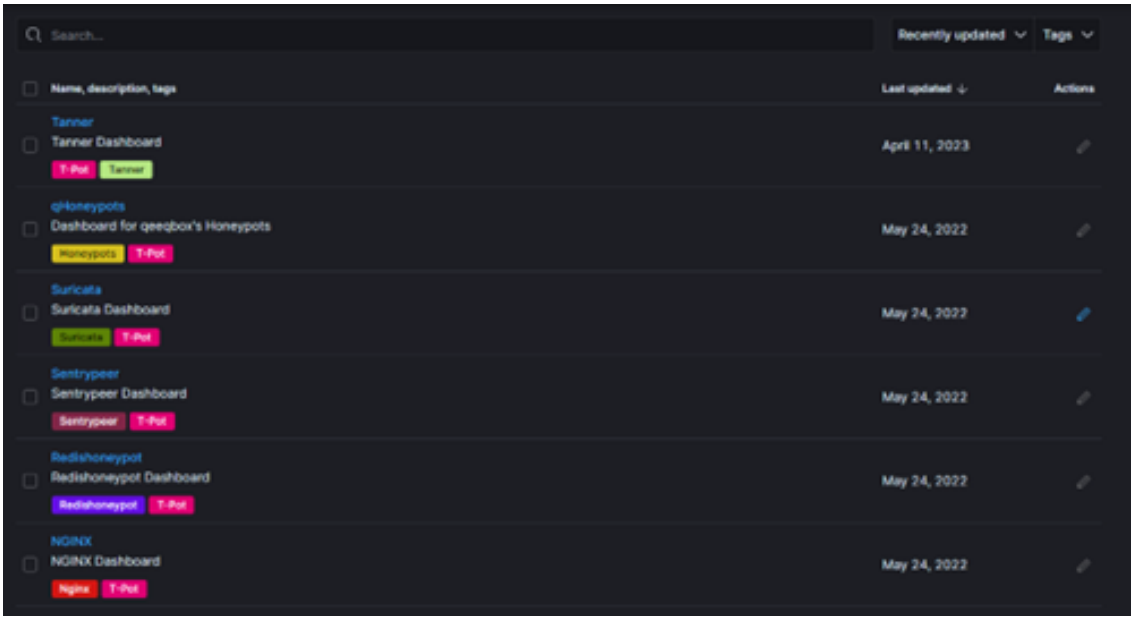

Figura 6: Conjunt de honeypots de T-Pot

Figura 7: Anàlisi general d'un honeypot

En les imatges anteriors es pot veure un exemple dels esdeveniments produïts dins de Suricata i un resum de les dades segons el tipus de dada. El que es mostra és el resum de tots els elements JSON però donant-li un format més visual, com per exemple el mapa. D'aquesta manera es pot veure millor des de quins països i quants cops hem estat atacats . A més, aquestes visualitzacions poden ser modificades com es desitgi, encara que per defecte, les gràfiques i visualitzacions ja són prou completes.

Així és com està estructurat i funciona la plataforma T-Pot. Hi ha informació més detallada a la seva pàgina de Github explicant els ports que es fan servir,

carpetes de dades, còpies de seguretat, etc. Per aquest projecte només s'ha centrat en la instal·lació i pujada de T-Pot a un servidor aïllat, i la recopilació i obtenció de les dades, però T-Pot va més enllà i permet la integració amb programes de tercers com servidors Apache, AWS o Dropbox, encara integrar-ho amb aquestes eines quedava fora del plantejament principal del projecte.

5.2 MISP i Cortex

MISP i Cortex són dues eines prou utilitzades en l'àmbit de la seguretat informàtica per millorar la capacitat de detecció i resposta d'amenaces. Les dues s'utilitzen per facilitar l'intercanvi i compartició de informació sobre incidents de seguretat permetent a les organitzacions o empreses col·laborar conjuntament, analitzar i respondre de la manera més eficient possible davant les amenaces que es puguin produir.

MISP (Malware Information Sharing Platform) és un programa de codi obert dissenyat per compartir informació sobre amenaces i malware, i proporciona un entorn col·laboratiu on els professionals de seguretat poden compartir indicadors de compromís (IoC) com adreces IP, noms de dominis, hashs, patrons de comportaments, etc. A més, MISP ofereix funcionalitats avançades com visualitzacions de relacions d'indicadors, que ajuden a detectar patrons ocults i comprendre millor les tàctiques, tècniques i procediments utilitzats per l'atacant.

Figura 8: MISP

De la mateixa manera, Cortex és un altre programa de codi obert d'anàlisi d'amenaces creat per TheHive Project que s'integra estretament amb MISP. Aquest programa proporciona una sèrie de serveis d'anàlisis automatitzats per a que ajudin als equips de seguretat a processar les seves dades, observables, etc. Cortex permet realitzar anàlisis de manera ràpida i eficient mitjançant l'ús de detectors, que són mòduls predefinits (o personalitzats) capaços d'identificar i analitzar patrons específics i activitat maliciosa. Aquests detectors poden ser utilitzats per detectar malware, IoCs, arxius, etc.

Figura 9: Cortex

Per tant, en conjunt, MISP i Cortex proporcionen una solució molt poderosa per la col·laboració i anàlisi d'amenaces. Amb aquests dos, els professionals de la seguretat poden treballar de manera més efectiva, compartint intel·ligència i actuant de manera proactiva per protegir els seus sistemes i xarxes contra amenaces actuals i emergents.

5.2.1 Configuració

Per instal·lar aquestes dues plataformes, s'ha executat sobre una màquina virtual de manera local amb el sistema operatiu Linux amb la distribució Ubuntu 20.04 LTS, i tant MISP com Cortex s'han instal·lat en el mateix Docker.

MISP s'ha configurat dins la màquina virtual en el port 80, i Cortex en el 9001, i on els dos programes proporcionen una pàgina web des d'on l'usuari els pot fer servir. Aleshores per entrar als programes s'ha de posar en el navegador l'adreça `192.168.1.200:<port>`. Com es pot veure, la màquina virtual es va configurar amb una IP privada i estàtica.

Tant per MISP com per Cortex, ens demanaran a l'inici unes credencials per iniciar sessió amb el nostre usuari. Per ambdós casos, s'ha de crear un usuari administrador, i una organització on pertanyarà aquest ja que és necessari a l'hora de compartir dades que ens mostri des de quina organització s'ha creat l'esdeveniment.

Un cop tot està instal·lat, falta integrar els dos programes. Per fer-ho, s'ha d'anar a la pàgina de Cortex i afegir un nou analitzador anomenat MISP_2_1:

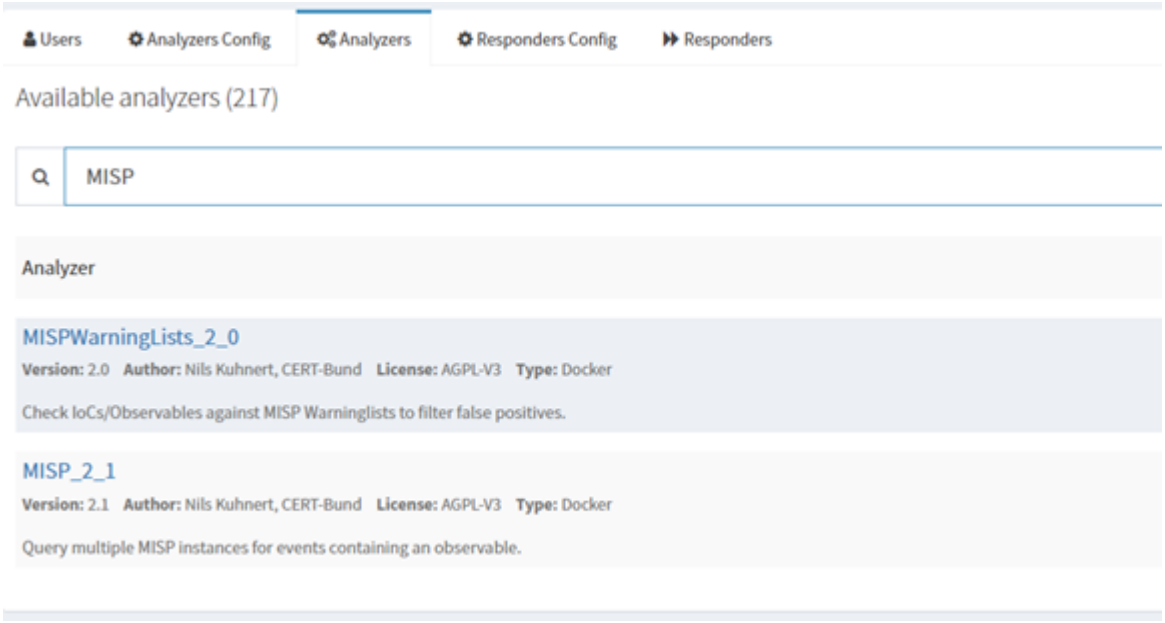

Figura 10: Analitzadors de MISP

En la imatge veiem dos analitzadors relacionats amb MISP. Pel projecte, s'ha utilitzat MISP_2_1 ja que ens permet buscar dins d'una instància MISP els esdeveniments o amenaces, i els observables que pertanyen a aquests esdeveniments.

Per configurar la integració, s'ha hagut d'afegir els camps de la IP de MISP, API key de l'usuari administrador per poder connectar-se a l'API de MISP, i indicar si es necessita comprovació de certificats: en el cas del projecte no es comprova ja que tot es fa en local:

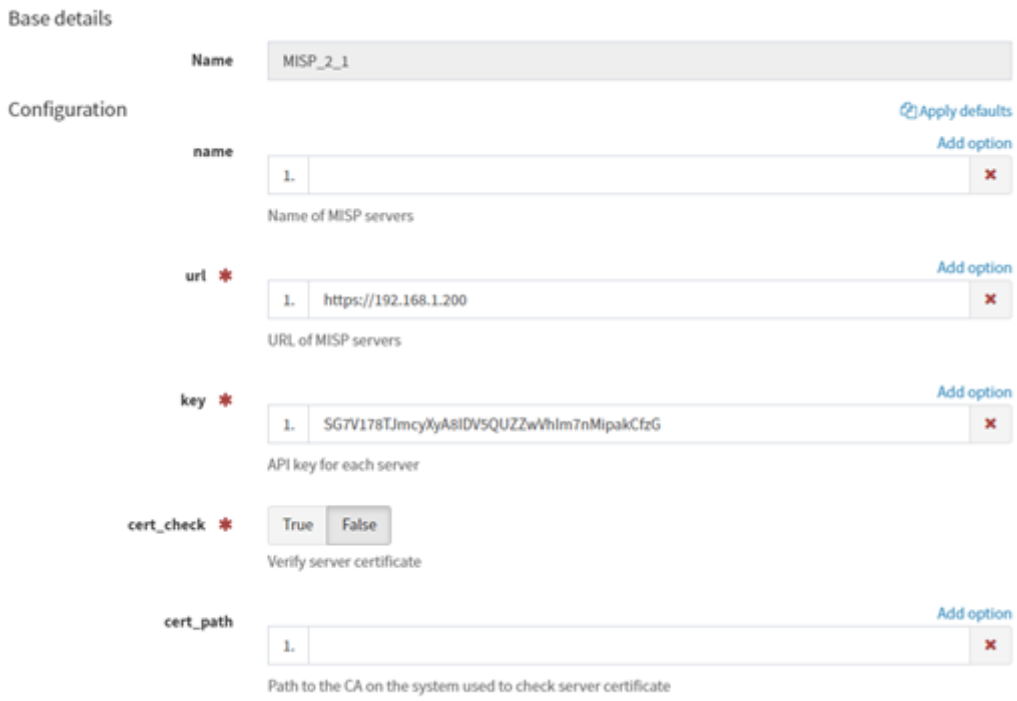

Figura 11: Configuració de MISP_2_1

Cortex ofereix molts més analitzadors com per exemple VirusTotal. L'avantatge d'utilitzar Cortex és tenir diversos analitzadors configurats per tal de veure si una IP o un hash d'un fitxer són detectats per varis analitzadors i assegurar que estem davant d'un incident de seguretat.

5.2.2 MISP

A continuació, es veurà com funciona MISP explicant quins són els seus elements principals, i després es mostrarà un exemple real amb el que es podrà entendre el seu funcionament.

MISP és una plataforma que permet la compartició d'amenaces de forma gratuïta degut a que és un programa de codi obert. Aquesta eina permet recopilar informació de qualsevol de les teves fonts i que, posteriorment, processa aquesta informació aplicant diferents mètodes com la normalització o la correl·lació. També permet la col·laboració de diferents equips, organitzacions, comunitats, etc, amb la qual notifica a les diferents parts quan hi ha una nova amenaça.

Quan es vol afegir una amenaça, s'ha de crear el que s'anomena un **Event**. Aquest element permet encapsular informació relacionada en un context determinat,

i és aquest el que utilitza MISP per compartir les amenaces entre equips. Aquests Events poden tenir diferents tipus d'altres elements de MISP com els Attributes, MISP Objects, Event Reports, entre d'altres.

Un **Attribute** és el bloc més bàsic d'informació que hi ha a MISP. És una dada concreta que pot utilitzar-se com a indicador de compromís o com a suport d'una altra dada. Uns exemples de Attribute poden ser Dominis, IP, URL, SHA1, etc.

Els **MISP Objects** els podem entendre com a grups d'Attributes que estan relacionats entre sí. Aquests elements s'utilitzarien, per exemple, per recopilar informació d'un fitxer, persona, dispositiu, etc.

Si volem afegir informació més detallada sobre el Event creat, podem utilitzar els **Event reports**. Aquests elements permeten afegir més informació que ajudi a descriure l'Event.

Com s'ha pogut veure, aquests tres elements permeten recopilar tot tipus de informació relacionada amb una amenaça. Tot i que amb aquests elements ja podríem generar intel·ligència, MISP permet aportar més qualitat a la informació donant a l'usuari l'opció d'afegir un context l'Event. Aquest context es pot afegir amb el que s'anomena **Taxonomies** o **Event graph**: les Taxonomies són etiquetes que se li poden afegir l'Event que ajuden a l'usuari a l'hora de poder filtrar les amenaces guardades dins de MISP, i l'Event graph és una visualització que té el propi Event on es pot veure la relació que hi ha entre els diferents elements. Això permet entendre molt millor com ha estat l'amenaça i quins patrons ha seguit l'atacant.

Exemple Ara es mostrarà un exemple real de com es podria utilitzar MISP, seguint unes bones pràctiques per una analista de ciberseguretat i utlitzant tots els elements explicats. Suposem que rebem el següent correu i nosaltres, com analistes, hem de recopilar la informació de l'atac:

Estimat/da xy,

Recentment hem patit un intent fallat de spearphishing dirigit al nostre CEO, amb els següents detalls:

El nostre CEO va rebre un correu electrònic el 13/09/2022 a les 15:56, que contenia un missatge personalitzat sobre la targeta d'informe de la seva filla. L'atacant es va fer passar per treballador de l'escola de la filla i va enviar el correu des d'una adreça falsificada (john.doe@luxembourg.edu). John Doe és un professor de l'estudiant. El correu electrònic es va rebre des de throwaway-email-provider.com (137.221.106.104).

El correu electrònic contenia un fitxer maliciós (adjunt) que intentaria descarregar una càrrega secundària des de https://evilprovider.com/this-is-not-malicious.exe (també adjunt, amb la IP 2607:5300:60:cd52:304b:760d:da7:d5). Sembla que la mostra intenta aprofitar la vulnerabilitat CVE-2015-5465. Després d'un breu anàlisi preliminar, la càrrega secundària té un C2 amb codi dur en https://another.evil.provider.com:57666

(118.217.182.36), al qual intenta exfiltrar credencials locals. Fins aquí hem arribat. Si us plau, tingueu en compte que aquesta és una investigació en curs. Voldríem evitar informar l'atacant de la detecció i us demanem amablement que utilitzeu només la informació continguda per protegir els vostres constituents.

Cordialment,

Quan es crea un nou Event, el programa demana unes dades inicials com: títol del Event, estat, organització, etc:

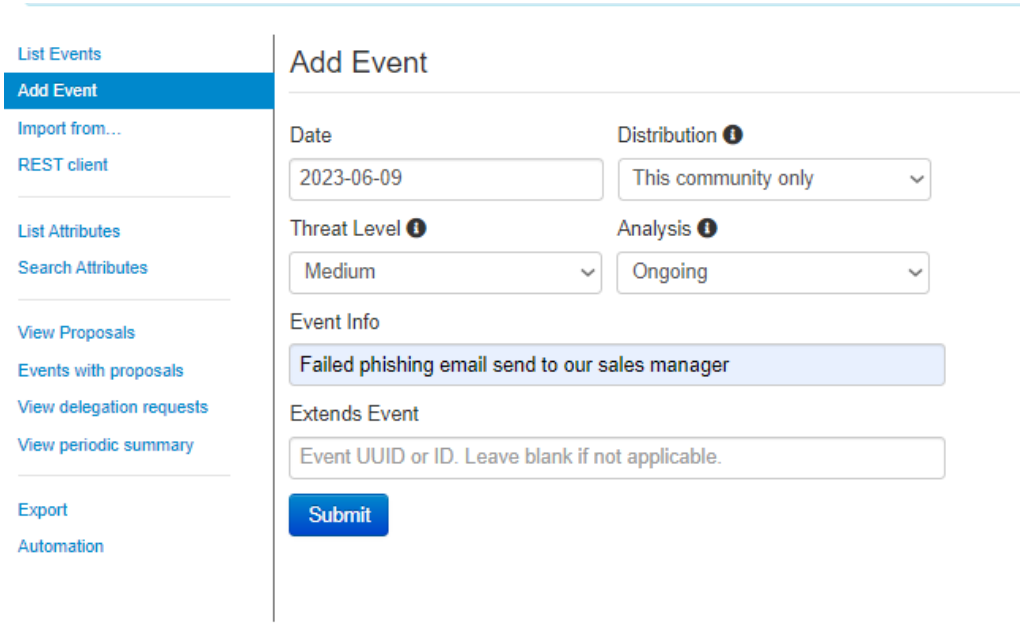

Figura 12: Dades inicials del Event

Quan es crea l'Event, MISP redirigeix a l'usuari a la pàgina que mostra els detalls d'aquest. I és en aquesta pàgina on es crearan tots els Attributes, Objects, etc. Per començar, se li afegirà un context inicial dient que aquest Event està en curs:

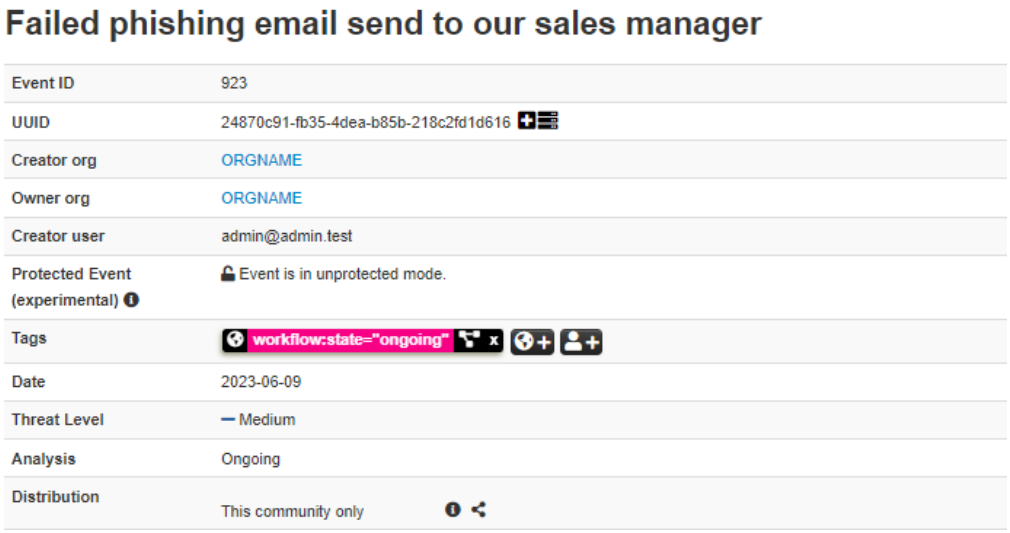

Figura 13: Context inicial

Un cop aplicat el context inicial, es començarà a afegir la informació rellevant de l'atac. Per començar, es guardarà la víctima, que en aquest cas és el professor de l'escola. MISP té plantilles d'objectes que ja contenen un conjunt d'atributs que es poden afegir. Com que es vol afegir la informació referent a una persona, s'utilitzarà el MISP Object Person, on es pot posar el nom, cognom i correu:

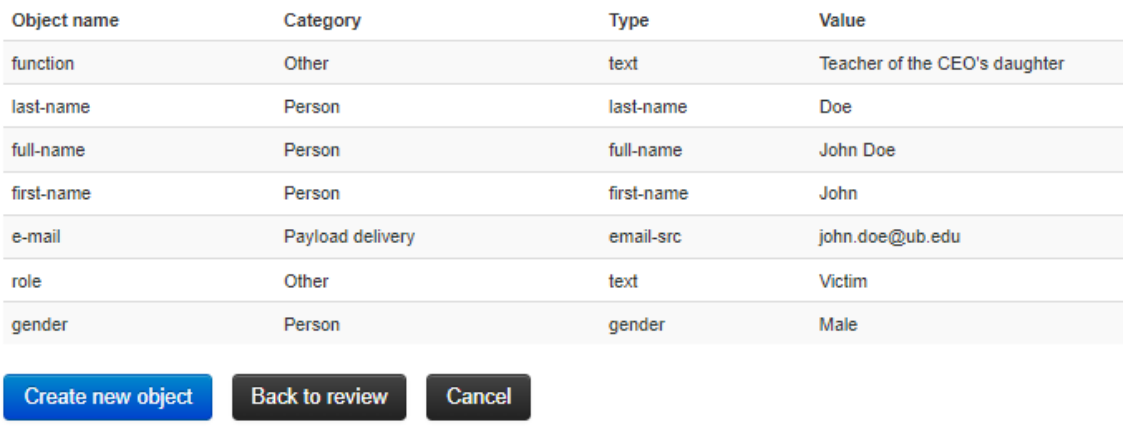

Figura 14: Informació de la victima en relació al correu

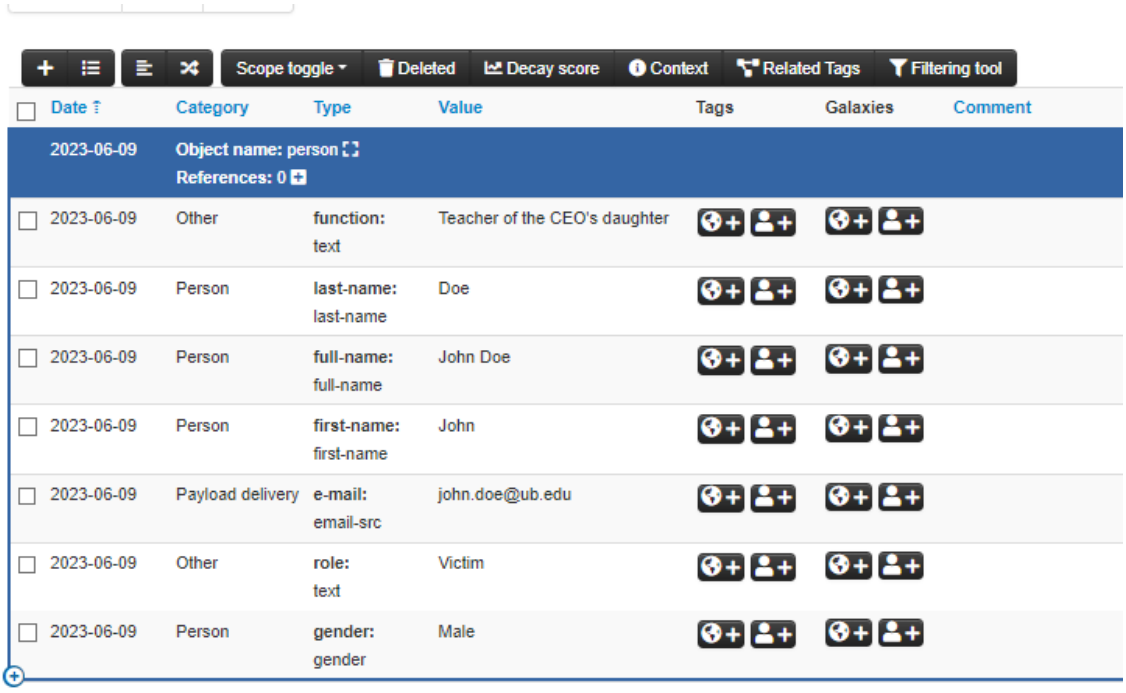

Figura 15: MISP Object creat dins del Event

D'aquesta forma és com es creen els objectes dins de MISP i s'extreuen les dades de l'atac. En aquest exemple també envien fitxers adjunts fent referència als programes que l'atacant volia executar. MISP també permet emmagatzemar aquests programes de manera segura fent servir el que s'anomena **Attachments**:

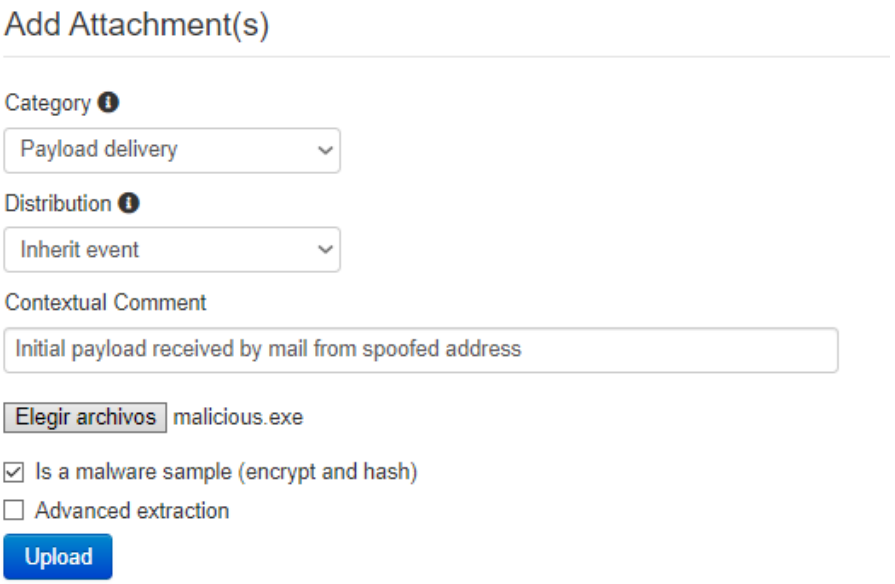

Figura 16: Programa maliciós recopilat

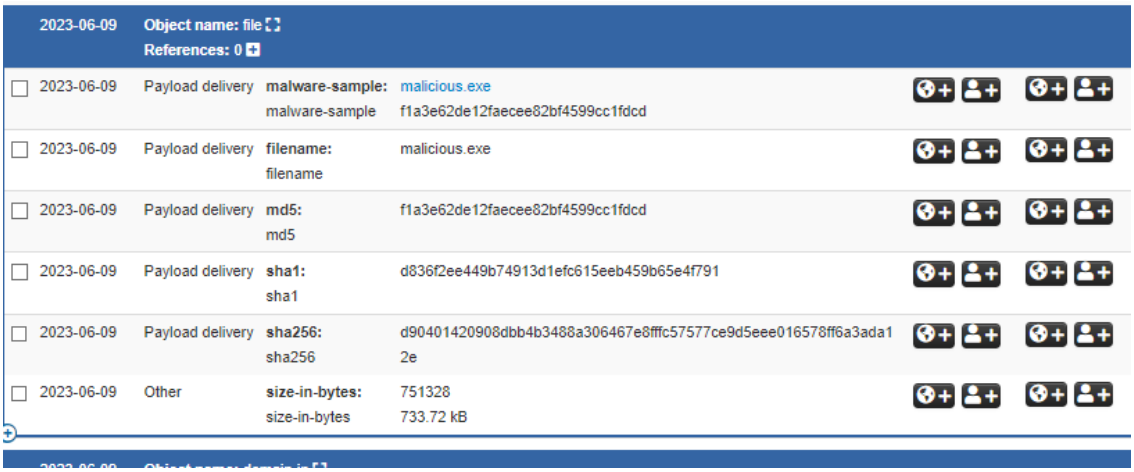

Figura 17: Programa maliciós convertit a MISP Object

Com es pot veure, un cop es crea el Attachment es converteix de forma automàtica a un MISP Object amb les dades relacionades del fitxer com SHA1, MD5, nom del fitxer, etc.

En el cas que es trobi una dada aïllada, una que aporta informació sobre l'atac però es pugui relacionar amb altres dades, s'haurà de crear un Attribute enlloc d'un MISP Object. En l'exemple es diu que l'atacant intenta aprofitar-se de la vulnerabilitat CVE-2015-5465 i, com que aquesta dada no se la pot relacionar amb cap altra, s'afegirà un nou Attribute a l'Event:

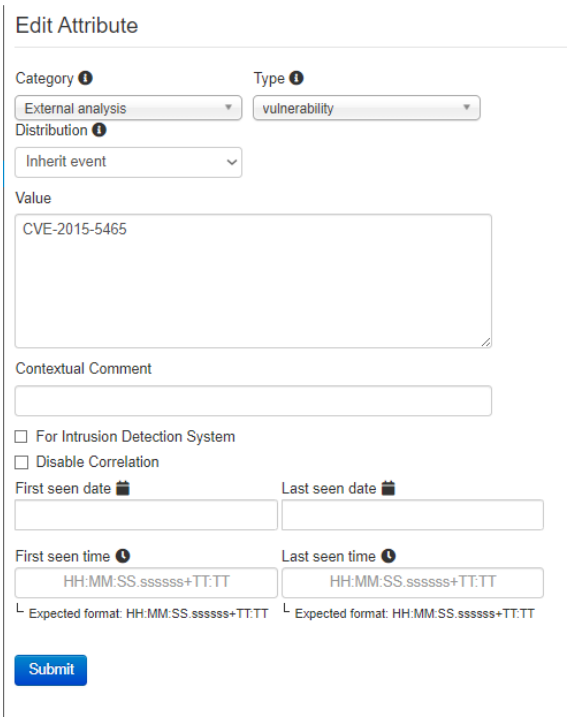

Figura 18: Camps d'un Attribute

A l'hora de crear MISP Objects, es pot donar el cas de que l'Object que s'estigui omplint comparteixi moltes dades amb altres que ja estiguin afegits. MISP detecta aquestes coincideències i avisa a l'usuari si es vol revisar l'objecte creat per tal d'adjuntar dos objectes en un i estalviar-se duplicar informació:

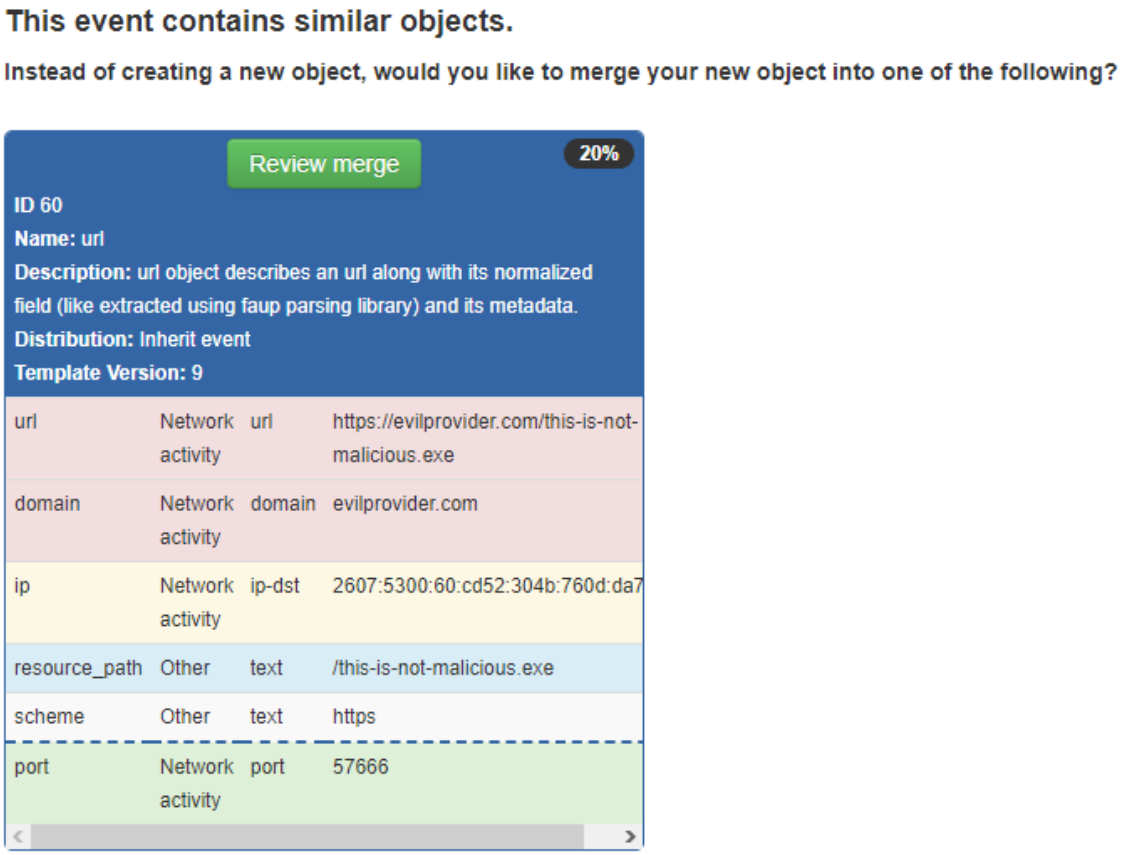

Figura 19: Detecció de relació entre MISP Objects

Ara es passarà a afegir més informació sobre l'atac. Com s'ha dit explicat anteriorment, existeixen els Event reports que fan la funció d'afegir informació addicional. En aquest exemple es podria utilitzar el contingut del correu en sí com a informació extra.

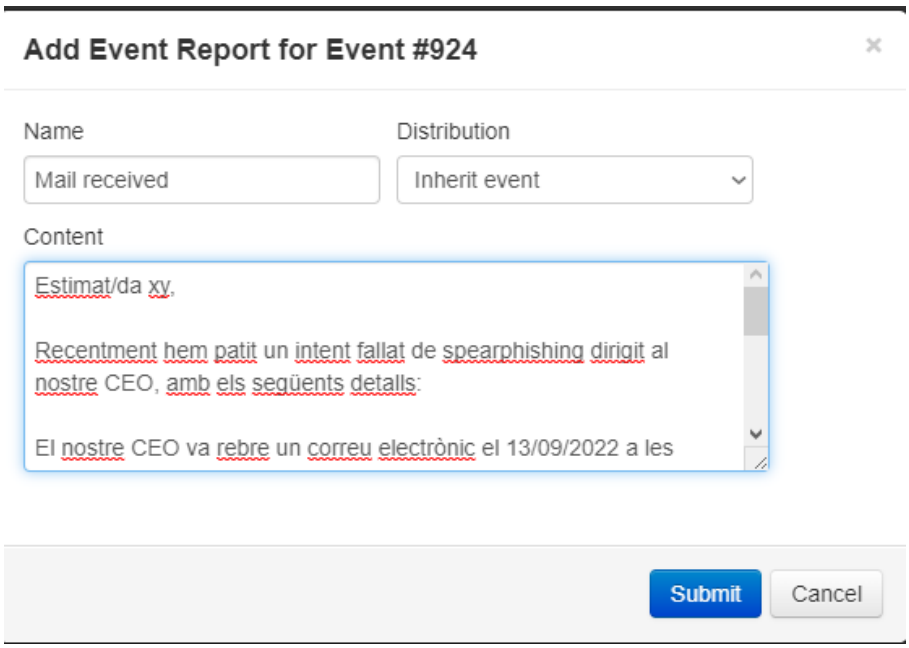

Figura 20: Event report de l'atac

Un cop tota la informació rellevant sobre l'amenaça s'ha afegit, és bona pràctica definir el context de l'atac, i com s'ha explicat abans, el context es pot afegir amb etiquetes o amb l'Event graph. Ara es passarà a explicar com funciona l'Event graph i quina funcionalitat té.

Tot i tenir la informació de l'atac guardada, en el cas de recopilar un gran volum de dades podria ser confús veure quins passos ha seguit l'atacant i els patrons o eines utilitzades. MISP dona una solució a aquest problema amb l'Event graph, permetent a l'usuari tenir una representació visual sobre l'atac. Aquesta eina funciona de manera que permet relacionar tots els MISP Objects i Attributes guardats. Seguint l'exemple, l'Event graph que sortiria inicialment seria el següent:

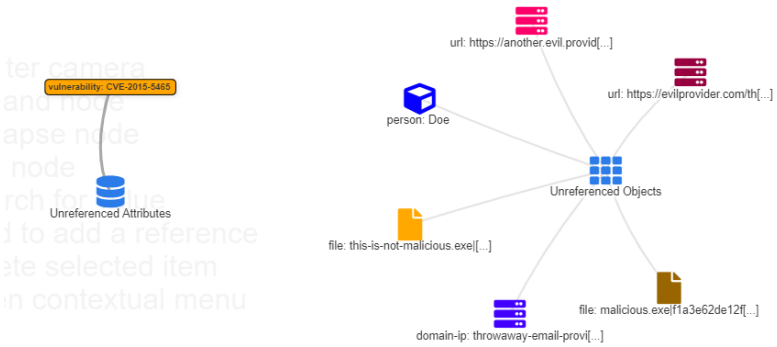

Figura 21: Event graph inicial

Com es pot veure, no hi ha relació entre els diferents elements. Pels analistes de seguretat informàtica és important recopilar les dades de manera precisa, ja que això pot portar a produir una resposta incident errònia si la intel·ligència generada és pobre. Una forma d'aplicar relació entre elements és com es mostra a la següent imatge:

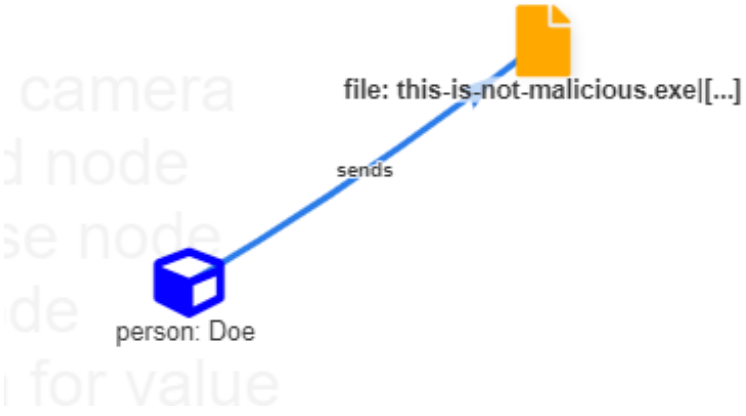

Figura 22: Exemple de relació entre elements

D'aquesta manera es pot veure que la persona John Doe ha enviat un fitxer maliciós, i afegint la relació anterior es pot enriquir molt més les dades, i a més entendre molt millor el context de l'incident. En aquest exemple, la relació completa de tots els elements seria la següent:

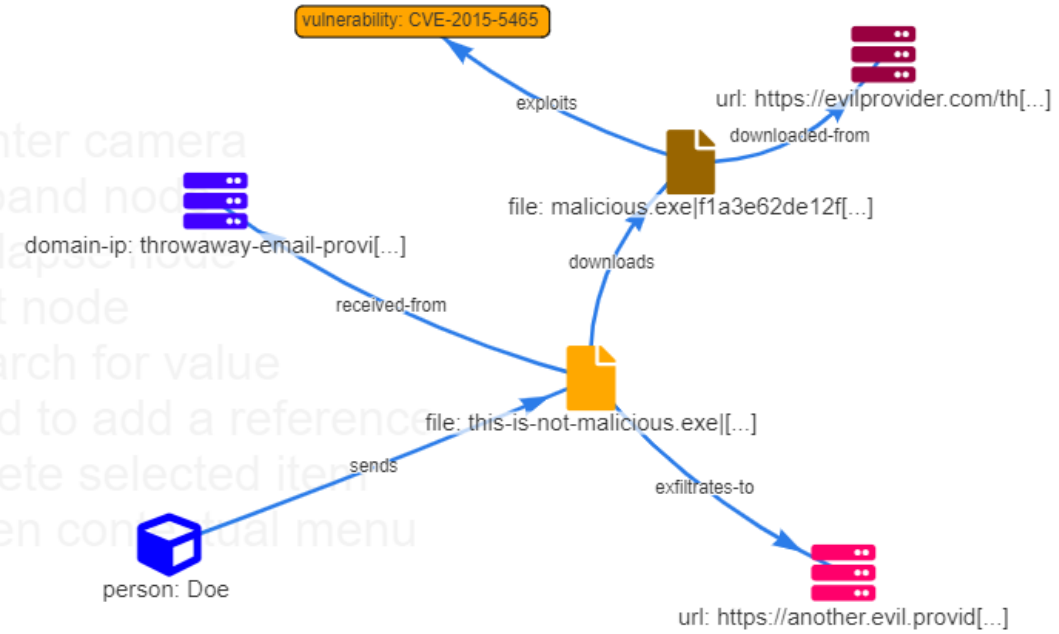

Figura 23: Relació entre elements complet

Per concloure amb MISP, s'ha vist que és una eina molt potent a l'hora de recopilar informació sobre incidents i amenaces. Amb aquesta plataforma es pot seguir vàries etapes del Cicle d'intel·ligència com s'han explicat anteriorment les fases de recopilació, processament i difusió.

Per altra banda, MISP redueix la càrrega de treball dels propis analistes de seguretat informàtica, ja que permet la col·laboració i compartició de les incidències. Aquesta opció els ajuda evitant que els analistes o equips recopilin la mateixa informació i tothom faci la mateixa feina.

5.2.3 Cortex

Ara es passarà a veure quin paper juga Cortex i com es complementa amb MISP. Per explicar el funcionament de Cortex s'utilitzarà l'analitzador MISP_2_1 que s'ha configurat prèviament.

Després de recopilar totes les dades a MISP, es pot utilitzar Cortex per tal d'analitzar si un indicador de compromís és realment maliciós, o pot comportar algun perill. Per analitzar aquestes dades s'ha d'omplir primerament la següent informació:

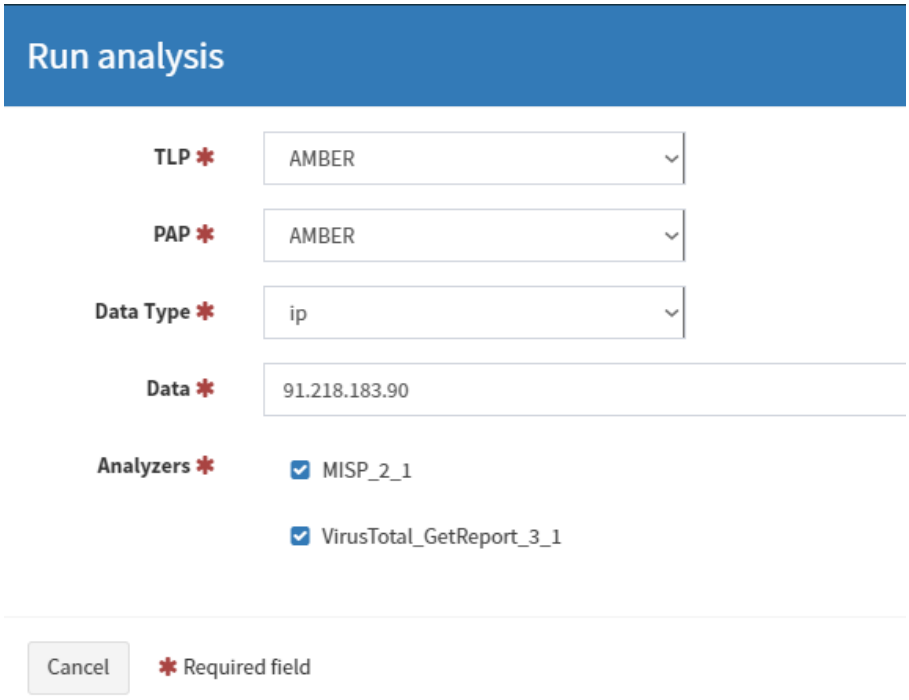

Figura 24: Anàlisi d'una IP amb Cortex

Dins d'aquest formulari es defineix el tipus de dada que es vol analitzar i s'escullen els analitzadors que es volen utilitzar. En aquest exemple es vol analitzar una IP amb els analitzadors de MISP i de VirusTotal. A més, afegir diferents analitzadors ajuda a verificar si realment aquella dada pot comportar alguna amenaça.

Un cop Cortex analitza les dades, retorna la resposta en format JSON. Seguint amb aquesta IP, els resultats que s'obtenen de MISP i VirusTotal són els següents:


```
{
  "summary": {
    "taxonomies": [
      {
        "level": "suspicious",
        "namespace": "MISP",
        "predicate": "Search",
        "value": "1 event(s)"
      }
    ]
  },
  "full": {
    "results": [
      {
        "url": "https://192.168.1.200",
        "name": null,
        "result": [
          {
            "id": "919",
            "orgc_id": "4",
            "org_id": "1",
            "date": "2023-04-13",
            "threat_level_id": "1",
            "info": "QUARTERRIG - Malware Analysis Report",
            "uuid": "04e8bb1e-b445-40a6-a68a-1ce85e32d229",
            "timestamp": "1682166784",
            "extends_uuid": "",
            "protected": null,
            "Orgc": {
              "id": "4",
              "name": "CIRCL",
              "uuid": "55f6ea5e-2c60-40e5-964f-47a8950d210f",
              "local": false
            },
          }
        ]
      }
    ]
  }
}
```

Aquest és un exemple de quin seria el JSON que es retornaria en el cas de l'analitzador MISP_2_1. Això voldria dir que podria ser una amenaça potencial degut a que s'ha trobat una amenaça registrada com Event on aquesta IP estava implicada. Per corroborar si la IP és maliciosa es comprovarà si VirusTotal també l'ha detectat com una amenaça. El JSON de VirusTotal és el següent:


```
{
  "summary": {},
  "full": {
    "type": "ip_address",
    "attributes": {
      "regional_internet_registry": "RIPE NCC",
      "jarm": "2ad2ad0002ad2ad0002ad2ad2ad2ade1a3c0d7ca6ad8388057924be83dfc6a",
      "network": "91.218.183.0/24",
      "last_https_certificate_date": 1681431024,
      "tags": [],
      "country": "GB",
      "last_analysis_date": 1685356619,
      "as_owner": "Evoxt Enterprise",
      "last_analysis_stats": {
        "harmless": 59,
        "malicious": 6,
        "suspicious": 1,
        "undetected": 21,
        "timeout": 0
      },
      "asn": 149440,
      "whois_date": 1685356621
    }
  }
}
```


Com es pot veure, hi ha sis analitzadors de VirusTotal que han detectat aquesta IP com a maliciosa. Aleshores es podria confirmar que aquesta IP és un perill potencial.

En aquest projecte s'ha interectuat amb MISP i Cortex des de les pròpies plataformes. Per aquest motiu, existeix el programa SIEM de TheHive Project que integra els dos programes en una mateixa plataforma, amb l'avantatge de poder analitzar les dades a temps real, i també proveeix visualitzacions estadístiques sobre els Events que es van registrant.

Per aquest projecte no s'ha arribat a implementar el SIEM perquè no era un objectiu principal, però sí que seria una bona consideració de cara a possibles millores que es puguin dur a terme.

5.3 Programa d'integració

Després de veure tant T-Pot com MISP es passarà a explicar com s'ha implementat el programa d'integració que extreu les dades d'Elasticsearch de T-Pot i les envia a MISP per recopilar-les i analitzar-les.

El següent pas un cop instal·lades les dues plataformes, és crear la connexió entre elles. Per això, es va investigar si ja existia una eina, programa o plug-in que permetés fer-ho automàticament, però no es va trobar cap resposta. La solució passava per crear un programa que integrés les dades de T-Pot a MISP sense haver de crear manualment els milers d'atacs i esdeveniments que s'havien recopilat.

5.3.1 Objectiu

L'objectiu principal del programa és que capturi els atacs dels diferents honeypots de T-Pot i exportar-los de tal manera que amb la API REST de MISP es puguin integrar de forma automàtica.

5.3.2 Requeriments i dependències

Per dur a terme aquest programa són necessàries, d'una banda, connectar-se a la base de dades de T-Pot i, de l'altra, connectar-se a l'API REST de MISP. Com que T-Pot utilitza Elasticsearch per guardar la informació dels atacs, la idea principal és connectar-se allà. Per fer-ho, s'ha utilitzat la llibreria Elasticsearch de Python, que permet connectar-se a una instància de Elasticsearch passant-li les credencials i la seva URL. Per a MISP, primer es va pensar en utilitzar la llibreria requests de Python i connectar-se a l'API mitjançant els seus endpoints. Però MISP ja proporciona una llibreria pròpia que ajuda a accedir a una plataforma MISP via API REST anomenada PyMISP.

A més, també s'ha utilitzat la llibreria abc de Python la qual permet treballar amb classes abstractes i fer un codi més net i reutilitzable.

5.3.3 Estructura del programa

El programa conté dos mòduls que s'encarreguen de connectar-se amb les dues plataformes. En el cas del T-Pot, el mòdul s'encarrega de rebre les dades de l'Elasticsearch en format JSON, extreure aquelles dades que interessen, i guardar-les en diccionaris de Python. Un cop guardades, es passa aquesta informació a un adaptador de MISP que s'encarrega d'enviar-les a MISP en el format correcte per a que la seva API REST accepti les dades. El programa té un aspecte com el de la següent imatge:

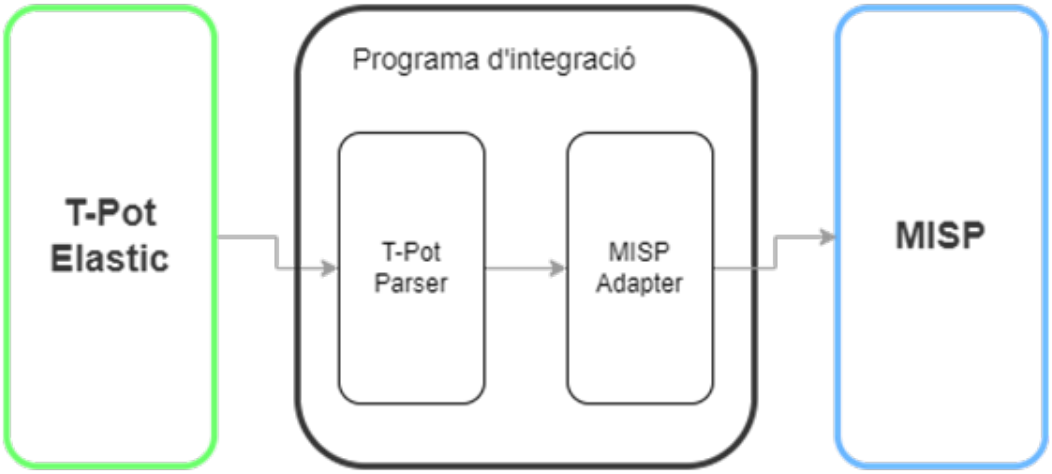

Figura 25: Esquema del programa d'integració

Però durant el desenvolupament del programa van sorgir algunes dificultats: un primer problema que va aparèixer va ser quan es volia exportar les dades, ja que no tots els honeypots enviaven la mateixa informació en els JSON. Això va impulsar el plantejament sobre com es podia fer la implementació de tal manera que es pogués reutilitzar el codi i fos escalable. La solució a la que es va arribar va ser implementar el patró de disseny Strategy, definint una estratègia que es basava en extreure els diferents atributs depenent del honeypot.

Amb la implementació d'aquest patró es va aconseguir que el codi fos adaptable a qualsevol canvi que pogués aparèixer, la qual cosa aplica el principi Open-Closed dels principis de SOLID. Funciona de manera que, per cada honeypot s'ha creat una classe que defineix les propietats que accepta del JSON i executa l'acció d'exportar les dades amb els paràmetres desitjats. En la següent imatge es mostra l'estructura que s'ha seguit amb els honeypots de Tanner, Suricata i Fatt:

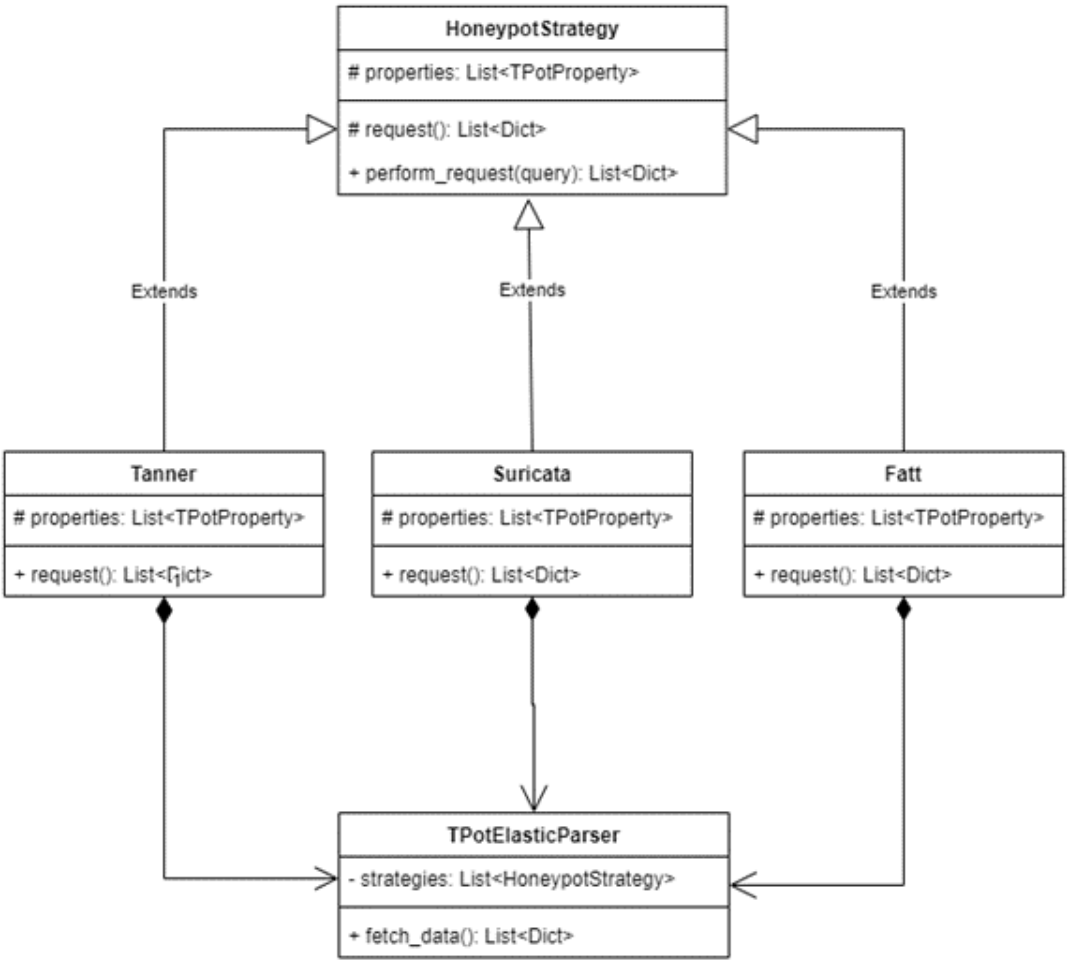

Figura 26: Diagrama UML de la integració amb T-Pot

En la imatge es pot veure clarament com funciona el patró Strategy implementat amb els honeypots Tanner, Suricata i Fatt.
Primerament es va definir la classe pare HoneypotStrategy, que conté una llista de propietats i dos mètodes. Aquesta llista de propietats és un atribut abstracte que les classes que heretin d'aquesta hauran de definir. El que s'intenta tenir en aquesta llista són aquells paràmetres del JSON que es volen extreure.
El mètode request() també és abstracte, i de la mateixa manera que les propietats, les classes que heretin l'hauran d'implementar. Totes les classes ho faran cridant el mètode perform_request(query), on aquest fa la crida a la base de dades d'Elasticsearch amb la "query" que es passa per paràmetre. Per exemple, en el cas de Tanner el mètode request() s'implementa de la següent manera:

```
def request():
    return self.perform_request("Tanner")
```

Aleshores, la classe TpotElasticParser és la que s'encarrega d'executar aquestes "strategies". Conté una llista de les diferents classes "strategy", i dins del mètode fetch_data() és on s'iteren per executar els corresponents mètodes request() i retornar els diccionaris amb les dades extretes.

També es va investigar sobre com es podia reutilitzar el codi a l'hora d'extreure les dades del JSON. És per això que es va aplicar el patró de disseny "Visitor" que permet fer l'extracció molt més modular, de tal manera que es poden reutilitzar les extraccions en els diferents honeypots. En el següent diagrama es mostra la implementació d'aquest patró:

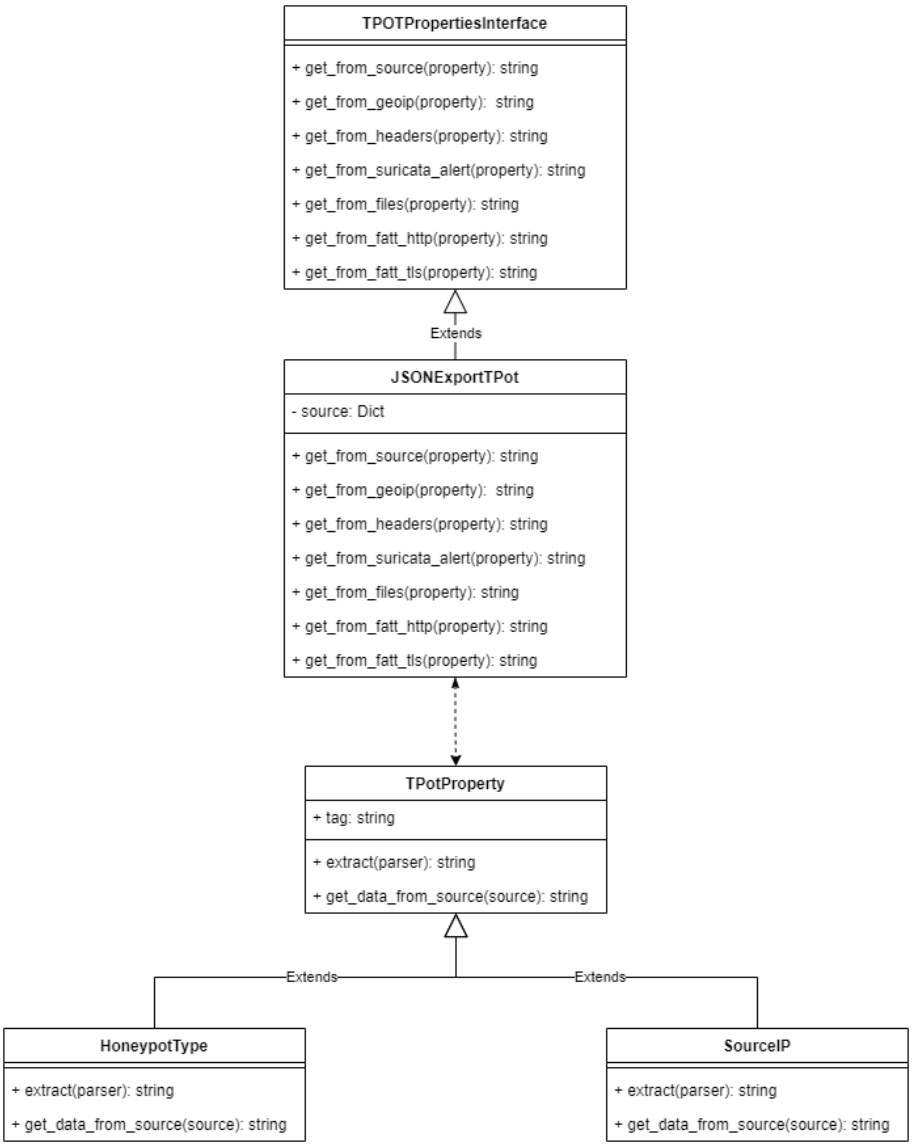

Figura 27: Diagrama UML de l'extracció de dades de T-Pot

Com s'ha vist abans, cada honeypot té una llista de propietats de tipus TPotProperty. Aquesta classe també és abstracta, i defineix dos mètodes: extract(parser) i get_data_from_source(source).

El primer mètode rep un paràmetre anomenat parser, el qual fa referència a la classe JSONExportTPot i on aquesta classe hereta de TPotPropertiesInterface en la que defineix un seguit d'accions que es poden fer a l'hora d'extreure les dades del JSON. I pel que fa al mètode extract(parser), l'única funció que realitza és cridar al mètode de JSONExportTPot que desitja.

L'atribut de la classe JSONExportTPot és el JSON que es rep de Elasticsearch, i aquest és el que es passa a cada propietat en el mètode get_data_from_source(source). Elasticsearch retorna el JSON en el format de diccionari de Python, aleshores dins d'aquest mètode només cal accedir al valor de la propietat del JSON com un diccionari normal de Python.

Per extreure les dades del JSON el que es fa és iterar totes les propietats que hi ha en un honeypot, a partir d'allà cridar el mètode extract(parser) de cada propietat i guardar el resultat dins d'un diccionari amb el tag com a clau de la parella.

Com s'ha esmentat abans, el mètode fetch_data() retorna una llista de diccionaris que tenen el següent aspecte:

```
{
    'type': 'Tanner',
    'country': 'Belgium',
    'dest-port': 80,
    'http-method': 'GET',
    'ip-reputation': 'known attacker',
    'src-ip': '104.199.31.214',
    'src-port': 52824,
    'timestamp': '2023-05-12T23:52:03.672224',
    'uri': '/',
    'user-agent': None
}
```

En aquest cas, podem veure una sèrie de dades relacionades amb un esdeveniment d'atac al nostre T-Pot. Aquest diccionari conté el tipus de honeypot on s'ha extret la informació, el país des d'on s'ha atacat, mètode HTTP utilitzat, IP de l'atacant, etc. Totes aquestes dades són les que exportarem a MISP per tal de tenir la recopilació de dades en un mateix lloc. Per exportar les dades a MISP es va pensar que la millor forma de fer-ho era aplicant el patró Adapter. Aquest patró ajuda a col·laborar dos objectes amb interfícies diferents. En el nostre cas es vol convertir les dades, en format diccionari, a objectes MISP per tal de guardar-les a la instància corresponent.

Per fer-ho, primer es va definir una classe MISP, que amb l'ajuda de la llibreria PyMISP permet a l'adaptador crear "Events" i atributs per a aquests per tal de que es pugui guardar la informació d'un mateix atac de manera conjunta. La classe té el següent aspecte:

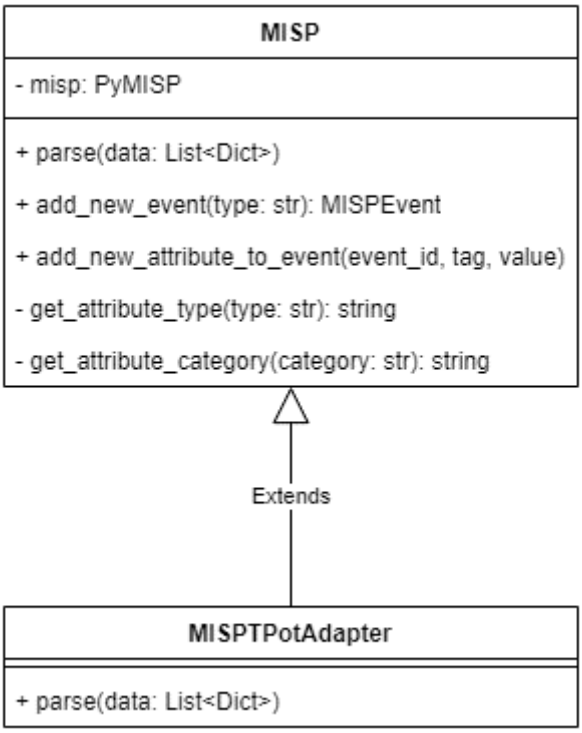

Figura 28: Diagrama UML de la connexió amb MISP

La classe MISP conté diversos mètodes que ajuden a l'adaptador a interactuar amb la instància. El mètode parse() és un mètode abstracte que implementa l'adaptador de manera que rep la llista de diccionaris per paràmetre i el que fa és cridar als mètodes de MISP per a cada diccionari.

Els mètodes privats get_attribute_type(type) i get_attribute_category(category) s'encarreguen de transformar dades que venen en el format de T-Pot a un altre format per a que MISP pugui acceptar els atributs d'un "Event". Per exemple, MISP, no té cap categoria o tipus d'atribut que s'anomeni "src-ip". Aleshores aquests mètodes s'encarreguen de classificar aquest atribut en una categoria "Network" i un tipus "ip-src". Així sí que acceptarà les dades i les guardarà l'"Event" corresponent.

5.3.4 Flux d'execució

Ara es mostrarà un exemple del flux d'execució que segueix el programa de manera resumida. Primer s'executa tota la part referent a les dades que provenen de T-Pot, es fa la petició a l'API i es va extraient les propietats dels JSON que es reben. Un cop es tenen totes les dades recopilades, es passa a la segona part cridant a l'adapter de MISP, encarregat de transformar les dades a un format per a que la instància de MISP pugui entendre-les i emmagatzemar-les:

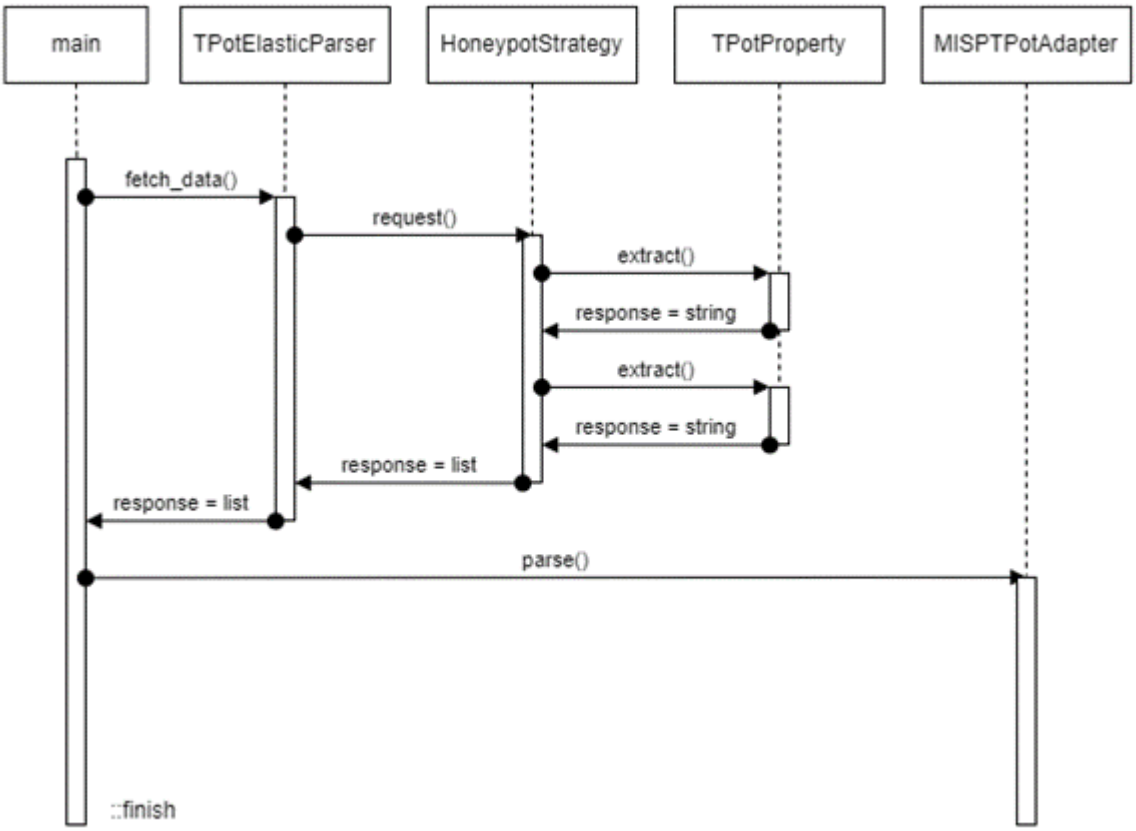

Figura 29: Diagrama UML de l'extracció de dades de T-Pot

Aquesta part té com a objectiu principal fer l'exportació de les dades de manera correcta fent ús de bones pràctiques a l'hora de desenvolupar un software. Cal dir però que hi ha millores a fer, com a exemple, veiem que el programa s'executa de manera seqüencial, la qual cosa obliga primer a fer un pas i després l'altre.

Per tenir un bon rendiment, es podria fer de manera asíncrona les dues funcions fent que un cop tinguem unes primeres dades de T-Pot, poguéssim emmagatzemarles a MISP de manera paral·lela a les següents peticions. D'aquesta manera aconseguiríem més velocitat. També un avantatge que tindria fer-ho així és que en cas que la connexió amb T-Pot es tallés a la meitat de l'exportació, ja tindríem unes dades emmagatzemades a MISP.

5.3.5 Securització del codi

Pel que fa al codi, el programa tracta amb dades sensibles com "l'API key" que es necessita per connectar-se a la instància MISP, o l'usuari i contrasenya per connectarse a l'Elasticsearch de T-Pot.

Hi ha diverses opcions per poder amagar aquestes dades per evitar que en el cas que facin enginyeria inversa, els atacants no poguessin accedir-hi. L'opció que s'ha

triat en aquest cas és guardar les dades sensibles a les variables d'entorn perquè és una manera ràpida, senzilla i segura d'implementar aquesta seguretat. Una altra opció que es va mirar també va ser implementar el programa demanant, primer de tot, el nom d'usuari i contrasenya tant per connectar-se a Elasticsearch com per MISP; i un cop es tingués l'usuari i contrasenya de MISP, fer una petició a través de HTTP per obtenir el token necessari per fer les crides a l'API.

Com bé es pot veure, la segona opció és molt més farragosa d'implementar, i també és menys segura degut a que existeix el cas que els atacants puguin interceptar la petició HTTP, i obtenir la informació si no s'ha gestionat bé la seguretat de les dades.

6 Despeses

La implementació d'aquest projecte amb les diferents eines implica la consideració de vàries despeses que s'haurien de tenir en compte. A continuació es mostrarà el llistat de despeses que pot comportar fer aquest projecte si es volgués extrapolar a una empresa real:

- Servidor Azure per la màquina virtual: T-Pot es troba virtualitzat a Azure, el que implica el cost de lloguer del servidor mensual i el manteniment de la pròpia màquina. Els costs varien depenent del tipus de recursos que es vulgui tenir per la instància de la màquina, però T-Pot requereix als sistemes uns mínims de capacitat i memòria per executar els honeypots. En el cas del projecte, s'ha escollit un pla de subscripció que comporta una despesa de 200€ al mes.

- Llicència de TheHive Project: Per una empresa seria una molt bona opció tenir les eines de recopilació i anàlisi d'amenaces en una mateixa plataforma. Com s'ha comentat, existeix TheHive que permet integrar MISP i Cortex en una mateixa plataforma. Hi ha el pla sense cost que té vàries opcions però està prou limitat. Després hi ha els plans de pagament com el "Gold" o el "Premium" que té un preu de 18.500€ i 24.500€, respectivament.

- Recursos humans: El projecte requereix personal expert en ciberseguretat i amb coneixements sobre les eines. Possiblement s'hauria de contractar una o vàries persones encarregades de gestionar els programes. A part, seria bo que es fes un seguiment i manteniment de codi per assegurar que el funcionament sigui òptim en cas d'actualitzacions d'aquestes eines.

- Temps i esforç: S'ha de tenir en compte també, que encara que no sigui una despesa quantitativa, també és important tot el temps de disseny, desenvolupament i implementació del programa. Depenent de la complexitat i la disponibilitat dels recursos, s'haurà de dedicar més o menys temps.

7 Resultats

Ara es passarà a explicar els resultats obtinguts per cada eina, i quines podrien ser les millores a fer per obtenir millors solucions tant de rendiment, visualització de dades o securització.

7.1 T-Pot

Amb T-Pot no es comptava amb el problema que cada trenta dies s'eliminava la informació dels honeypots. Això ho fa per no haver de guardar milers de dades dins de la màquina virtual i tenir espai per recopilar dades futures. Per exemple, en el primer mes després d'haver instal·lat la plataforma, en el honeypot Suricata es va arribar a recopilar més de 1.500.000 atacs.

Existia la possibilitat que quan T-Pot es pugés al servidor no cridés l'atenció a cap atacant, la qual cosa m'hauria obligat a buscar altres formes per atraure als atacants. Però com es veurà a continuació, s'ha pogut recopilar un gran conjunt de dades que ens han servit per poder examinar-les i emmagatzemar-les a MISP.

Tot i que s'ha pogut recopilar milers de dades amb T-Pot, no tots els honeypots han atret als atacants; de fet, només hi ha hagut quatre que ho han fet de manera regular. En les següents seccions es mostrarà un resum de les dades dels atacs recopilats pels honeypots Cowrie, Fatt, Tanner i Suricata des de que es va llençar:

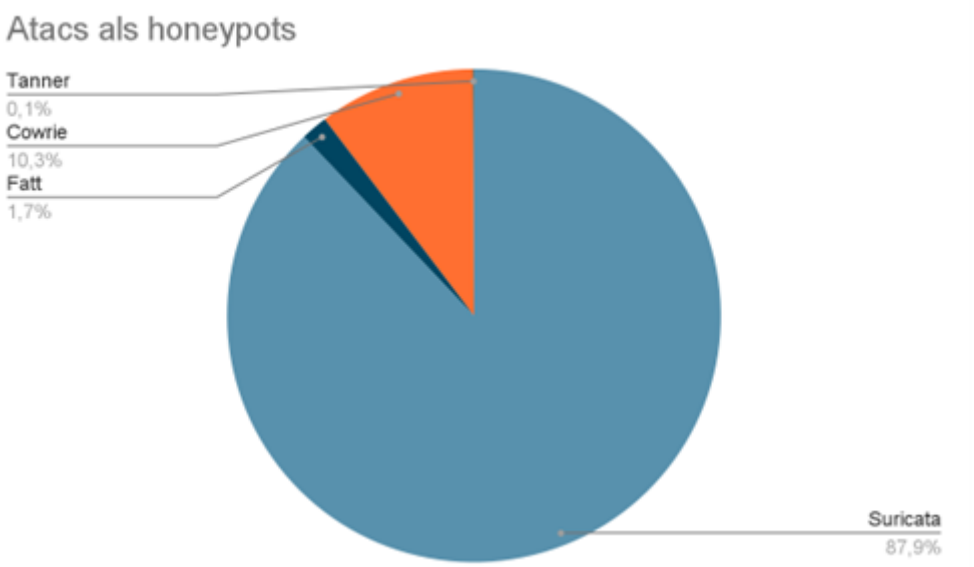

Figura 30: Total de atacs recopilats

A diferència de Suricata, els altres honeypots han rebut més o menys atacs, però tot i així, tant Fatt com Cowrie han superat els 100.000 atacs durant tres mesos.

7.1.1 Suricata

El honeypot Suricata ha estat el que més resultats ha obtingut: ha estat l'objectiu principal pels atacants amb aproximadament 5.000.000 atacs amb tres mesos. En la figura 34 es pot veure clarament que des de Hong Kong és des d'on s'han produït més atacs.

Figura 31: Total de atacs recopilats a Suricata

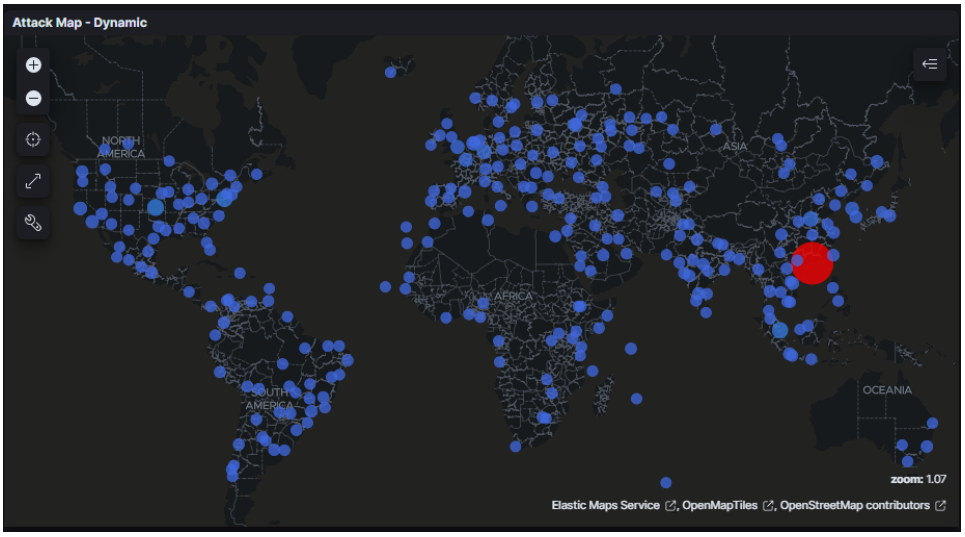

Figura 32: Mapa de la localització dels atacs de Suricata

T-Pot ofereix la possibilitat de filtrar les dades. En el cas que es volgués investigar més informació dels atacs de Hong Kong, T-Pot ajuda a fer aquesta investigació. Observant els atacs produïts, s'ha vist que la IP 168.63.129.16 és la que ha fet pràcticament el 100% dels atacs:

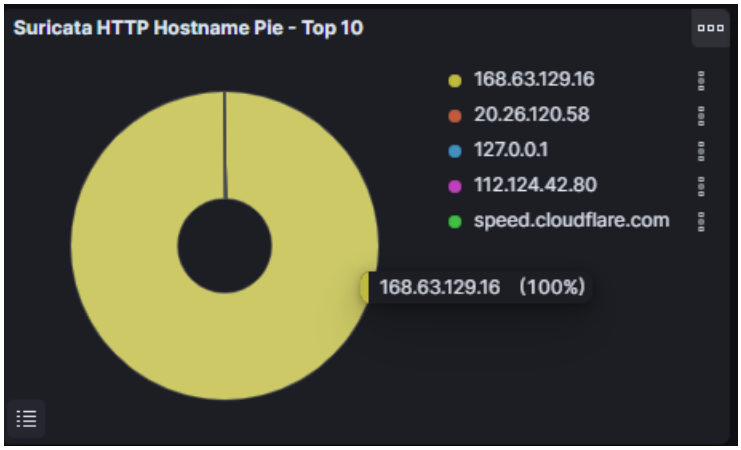

Figura 33: Cerca de IPs de Hong Kong

Si es volgués veure més informació sobre aquesta IP en concret, T-Pot redirigeix a l'usuari a la pàgina tallosintelligence.com de Cisco on es pot veure més informació relacionada amb aquesta IP:

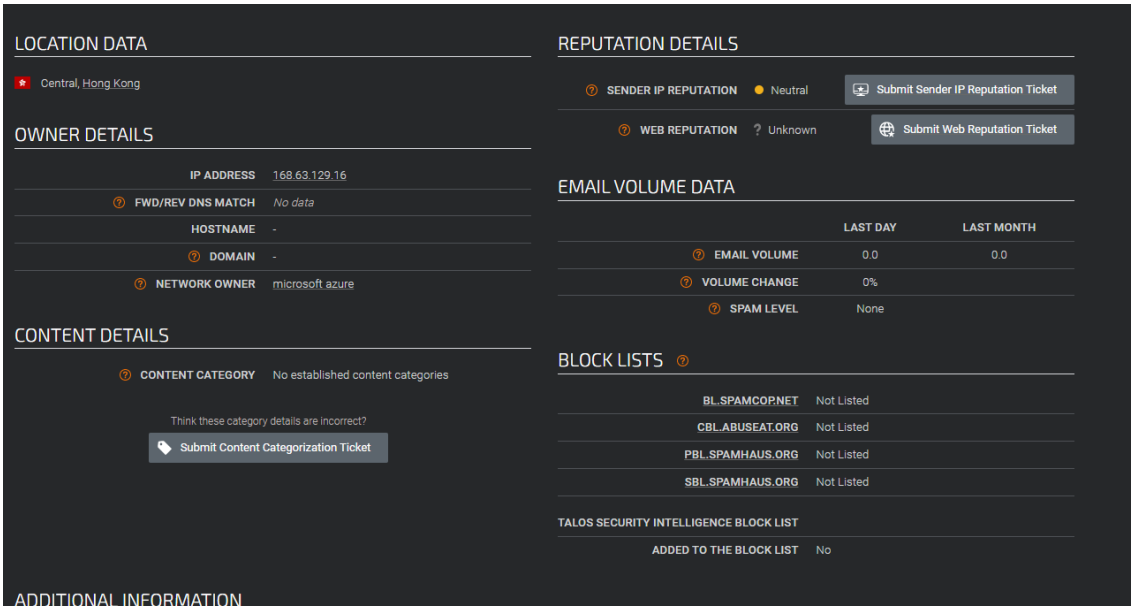

Figura 34: Cerca de IPs de Hong Kong

7.1.2 Cowrie

Cowrie ha obtingut bons resultats amb aproximadament 500.000 atacs recopilats. Aquest honeypot no s'ha utilitzat en el programa d'integració del projecte, però seria una bona font d'informació per extreure dades:

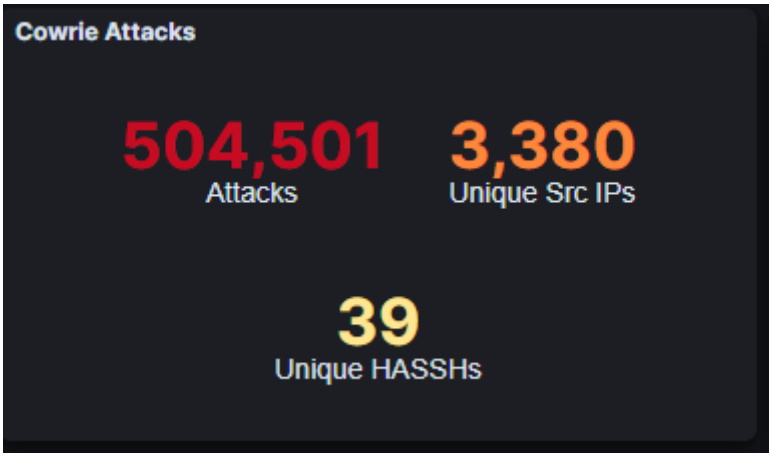

Figura 35: Total de atacs recopilats a Cowrie

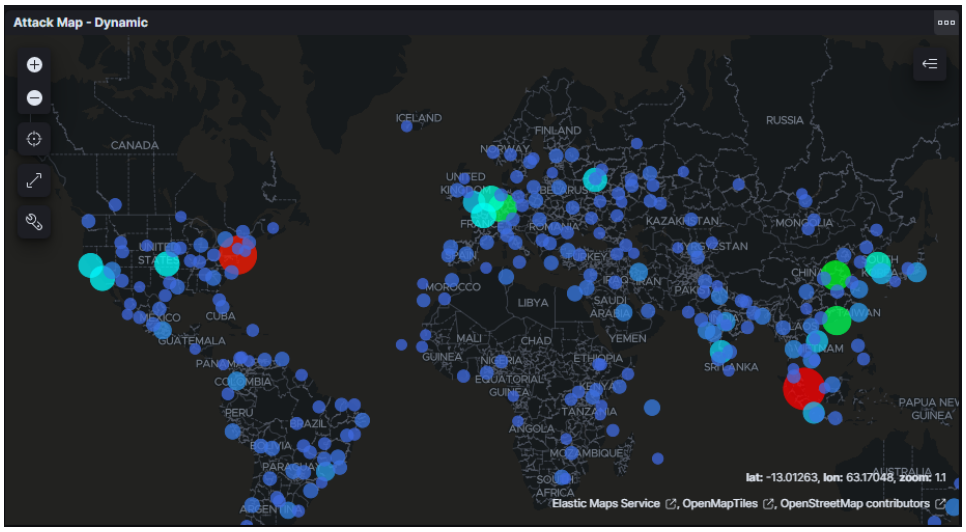

Figura 36: Mapa de la localització dels atacs de Cowrie

7.1.3 Tanner

En el cas de Tanner, aquest honeypot ha estat el que menys atacs ha rebut. Tot i així, s'ha pogut recopilar un conjunt significatiu de dades en relació als atacs:

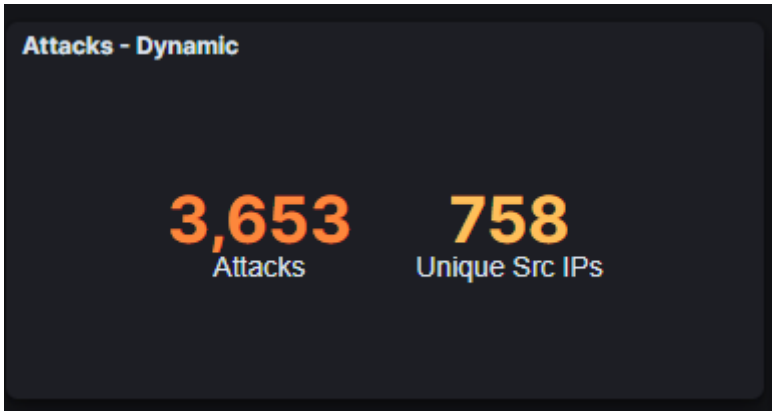

Figura 37: Total de atacs recopilats a Tanner

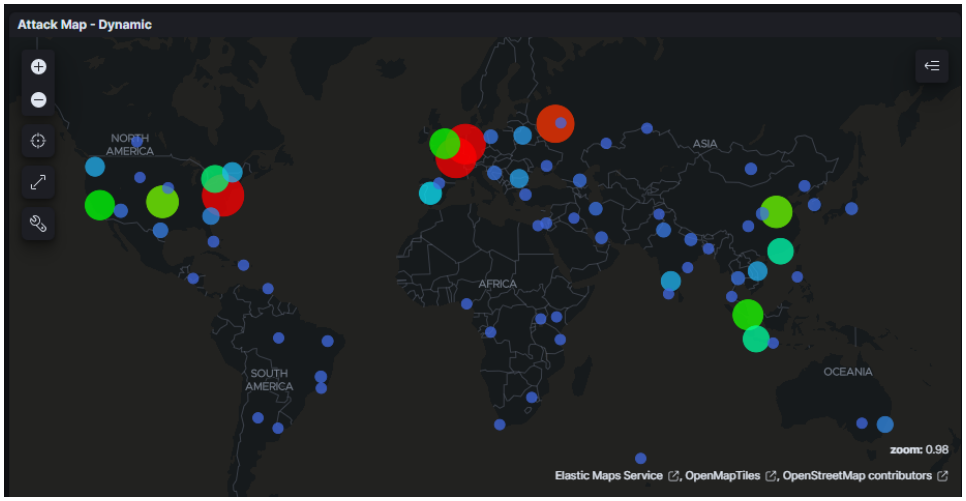

Figura 38: Mapa de la localització dels atacs de Tanner

7.1.4 Fatt

I en el honeypot Fatt, tot i no tenir tants atacs recopilats, li ha quedat poc per arribar als 50.000 atacs:

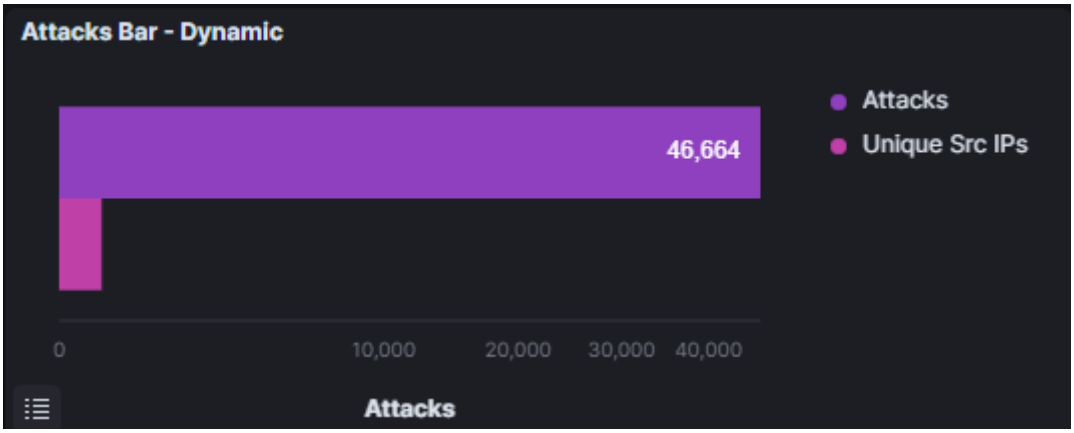

Figura 39: Total de atacs recopilats a Fatt

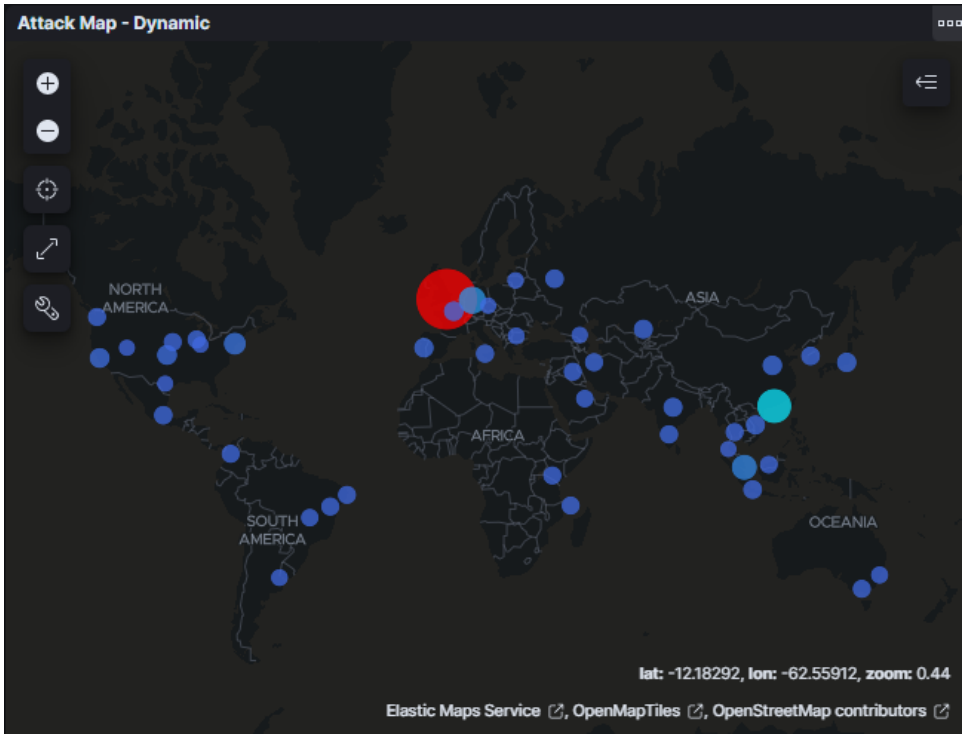

Figura 40: Mapa de la localització dels atacs de Fatt

7.2 MISP i el programa de integració

El programa d'integració ha pogut recopilar les dades de T-Pot i guardar-les a MISP amb èxit. Aquest seria un exemple d'atacs guardats a MISP amb el format d'"Events":

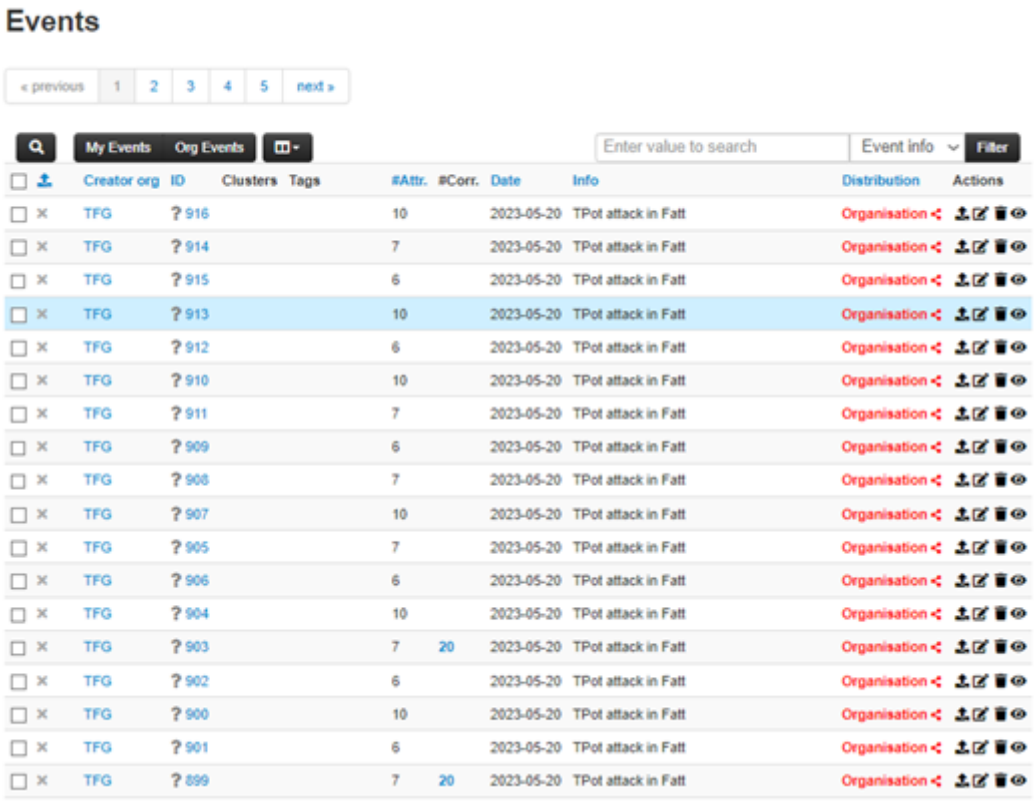

Figura 41: Resultat de la integració amb MISP

Quan es va fer la implementació del programa d'integració es va valorar fer-ho amb objectes, ja que es recomana guardar la informació amb aquest format, però fer-ho d'aquesta manera implicava augmentar el cost del programa degut a que primer, s'havien de guardar els atributs fent una crida a l'API, i posteriorment, guardar els mateixos atributs en format d'objecte fent una altra crida a l'API. Aleshores, es va optar per guardar la informació amb atributs ja que era més senzill i ràpid. Seria bo estudiar la manera d'optimitzar aquestes crides a API i poder guardar la informació en el format de MISP Object.

8 Possibles millores

El projecte implementat presenta un conjunt de millores que es poden dur a terme en un treball futur. Aquestes són les millores que crec que es podrien realitzar:

- Crides a les API asíncrones: Tant per la crida a l'API d'Elasticsearch, com a la crida de l'API de MISP, es podria optimitzar l'eficiència i rendiment del programa fent que les crides dels honeypots es facin de manera paral·lela.
- Extendre la recopilació de dades a més honeypots: El programa d'integració de moment només accepta tres honeypots dels 22 que hi ha. El fet d'afegir honeypots a nivell de codi no hauria de suposar cap problema de modificació ja que només s'hauria de crear una classe que heretés de HoneypotStrategy i declarar les seves propietats.
- Utilitzar objectes enlloc d'atributs: Com s'ha comentat abans, la informació que es guarda a MISP des del programa es fa amb atributs, la qual cosa quan es vol consultar un "Event" no es pot veure de forma clara la relació que hi ha entre les dades i dificulta la comprensió de l'amenaça.
- Ús de TheHive Project: Per un treball futur, seria una gran ajuda tenir tant MISP com Cortex integrats en una mateixa plataforma, de tal manera que es poguessin consultar les dues eines des d'un mateix lloc. Per tal de solucionar aquest aspecte TheHive Project permet fer aquesta fusió, i pot ajudar a comprendre de manera molt més clara les dades que es recopilen, ja que té components visuals que ajuden a l'usuari a entendre de forma més clara les dades amb les que treballa.
- Securització de T-Pot: Quan es va instal·lar T-Pot, no es va tenir en compte la seguretat del dispositiu en el qual estava instal·lat. Tot i estar en un entorn aïllat fora de la xarxa principal, no estaria de més afegir més seguretat a la màquina virtual per a que no ens la poguessin vulnerar en cas d'atac.
- Afegir una interfície gràfica al projecte: En aquesta primera versió, el programa no té cap interfície gràfica, sinó que s'executa des de consola. Per l'usuari que utilitzés el programa seria de gran ajuda tenir una interfície gràfica descriptiva amb les opcions que permetés fer el programa.
- Afegir context als "Events": A l'hora de la creació d'"Events" a MISP, no es contextualitzen. Una bona pràctica és afegir etiquetes o "tags" que permeten entendre aquella amenaça, i a part, ajuden a filtrar-les dins del llistat recopilat.
- Compartir les amenaces: MISP és una eina que permet compartir les amenaces recopilades. En aquest projecte com que no he implementat la generació d'intel·ligència amb objectes, i no he afegit context, he decidit no compartir aquestes dades ja que no aportarien el valor necessari per a la comunitat.

9 Conclusions

La idea des d'un inici del projecte va ser recopilar dades d'atacs i tenir una base de dades on es pogués analitzar els atacs i detectar patrons. Des de la desconeixença del tema es va investigar i estudiar des d'un marc més teòric els conceptes principals per la generació d'intel·ligència i la resposta incident.

Un cop amb una idea més clara del tema, s'ha buscat les eines necessàries per poder dur a terme aquesta base de dades. T-Pot, MISP i Cortex han estat les eines escollides i, amb la instal·lació d'aquestes, s'han pogut recopilar i analitzar un gran conjunt d'amenaces de forma automàtica.

Amb l'ajuda de T-Pot s'ha aconseguit atraure a molts criminals que han intentat vulnerar els diferents honeypots que tenia la plataforma. Això ha permès recopilar i detectar tota classe d'activitat maliciosa, de la qual s'ha obtingut una gran varietat d'indicadors de compromís com IP, hashs, localitzacions, etc. La instal·lació de MISP ha ajudat a entendre com generar intel·ligència de manera eficient així com comprendre més fàcilment les amenaces recopilades i els patrons que segueixen. Després s'ha integrat Cortex amb MISP, la qual cosa ha aportat més anàlisis a les dades recopilades, i així, permetre una gestió més eficient de les dades.

La solució a la qual s'ha arribat perquè MISP i T-Pot treballessin de manera conjunta ha estat la implementació del programa d'integració entre T-Pot i MISP. Aquesta part ha permès automatitzar correctament l'exportació de les dades de T-Pot cap a MISP. Un cop provat el programa s'ha pogut veure que el resultat de l'exportació ha funcionat, i també s'ha vist diverses millores que es poden dur a terme per tenir resultats més precisos a l'hora de generar intel·ligència i augmentar la velocitat d'exportació.

En resum, s'ha desenvolupat una solució a la detecció i recopilació d'amenaces, i a la generació d'intel·ligència de manera automàtica amb les eines de T-Pot, MISP i Cortex. Aquest projecte pot ajudar a prevenir incidències de seguretat i amenaces en empreses o organitzacions, ja que és capaç de recopilar i analitzar dades en poc temps.

10 Bibliografia

Referències

[1] Roberts, S. J.; Brown, R. (2017). Intelligence-driven incident response: Outwitting the adversary. "O'Reilly Media, Inc.". (Conslutat Desembre-2022)

[2] TheHive Project. TheHive, Cortex and MISP: How They All Fit Together: `https://blog.thehive-project.org/2017/06/19/thehive-cortex-and-misp-how-they-all-fit-together/`. (Consultat Gener-2023)

[3] MISP Thread Sharing. MISP Documentation and Support: `https://www.misp-project.org/documentation/`. (Consultat Gener-2023)

[4] Cortex. Cortex: `https://github.com/TheHive-Project/Cortex`. (Consultat Gener-2023)

[5] Medium. Installing MISP, TheHive and Cortex: `https://wmvalente.medium.com/installing-misp-the-hive-and-cortex-part-5-d8a21c886fa8`. (Consultat Gener-2023)

[6] ls111's Cybersecurity Blog. TheHive, Cortex & MISP Installation Using Docker Compose: Ep10: `https://ls111.me/thehive-cortex-misp-installation-using-docker-compose/`. (Consultat Gener-2023)

[7] Telekom Security. T-Pot - The All In One Multi Honeypot Platform: `https://github.com/telekom-security/tpotce`. (Consultat Febrer-2023)

[8] Elasticsearch. Elasticsearch Guide - REST APIs: `https://www.elastic.co/guide/en/elasticsearch/reference/current/rest-apis.html` (Consultat Març-2023)

[9] MISP Thread Sharing. MISP OPENAPI Spec: `https://www.misp-project.org/openapi/`. (Consultat Abril-2023)

[10] Freeman, E.; Robson, E.; Bates, B.; Sierra, K. (2004). Head First Design Patterns: A Brain-Friendly Guide. "O'Reilly Media, Inc.". (Consultat Maig-2023)

[11] Refactoring·Guru. Refactoring Guru: `https://refactoring.guru`. (Consultat Maig-2023)

[12] Medium. Keep your code secure by using environment variables and env files: `https://towardsdatascience.com/keep-your-code-secure-by-using-environment-variables-and-env-files-4688a70ea286`. (Consultat Maig-2023)