IntelOwl per analisi di IP in Blacklist

Manuel Ceroni <m.ceroni2@studenti.unipi.it>
Luca Ferretti <l.ferretti13@studenti.unipi.it>

Introduzione

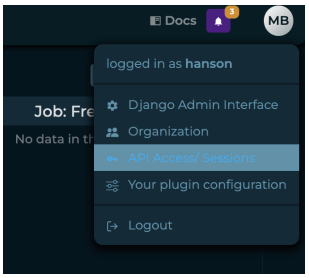
IntelOwl è un “Open Source Intelligence, or OSINT solution to get threat intelligence data about a specific file, an IP or a domain from a single API at scale”. Nello specifico abbiamo utilizzato il software per la gestione di IP sospetti in modo da avere un’ulteriore analisi per comunicazioni provenienti da indirizzi IP poco fidati.
Il nostro progetto apre un file PCAP, passato come argomento, e raccoglie gli IP delle comunicazioni al suo interno. Effettua un’iniziale scrematura con delle blacklist passate al programma che vengono scaricate all’inizio di ogni avvio e, gli indirizzi contenuti nelle blacklist, vengono inviati a IntelOwl tramite le apposite API fornite dagli sviluppatori.
In aggiunta, dopo aver comunicato gli indirizzi a IO, il programma rimane in attesa degli esiti e se un Job (quindi una delle analisi di IO) risulta con una pericolosità bassa o nulla, viene cancellato. In questo modo un amministratore potrà vedere nella dashboard a colpo d’occhio solamente gli indirizzi malevoli e potrà indagare sulle suddette comunicazioni.

Prerequisiti e Istruzioni

Per poter eseguire il programma è necessaria innanzitutto l’installazione di IntelOwl e la creazione di un account. Per l’installazione seguire le istruzioni nella documentazione fornita al seguente link https://intelowl.readthedocs.io/en/latest/Installation.html#tl-dr.

Attenzione: nelle ultime versioni di IntelOwl abbiamo riscontrato dei problemi nell'installazione, issue segnalate comunque da altri utenti. In caso che il problema non sia risolto provare una versione differente del programma

Una volta completata accedere al client per poter creare un account. Per poter funzionare il nostro programma ha bisogno della chiave di accesso quindi procedere con la creazione della suddetta.



La chiave così ottenuta va inserita nel file “pane_config.ini”.
A seguito di ciò procedere con l’installazione delle dipendenze, quindi eseguire

```
pip3 install pytricia pyshark urllib ipaddress configparser
argparse
```

è inoltre necessaria l’installazione di Wireshark, utilizzare il proprio gestore di pacchetti.

Viene fornito un file preliminare contenente degli indirizzi a file di blacklist, è possibile aggiungerne o rimuoverne a piacimento. Per il formato vedere le liste fornite.
Il programma ha bisogno di due parametri, uno che indica il path al file di blacklist e uno che indica il path al file PCAP che si vuole analizzare. Per l’esecuzione fare riferimento al comando help :

```
python3 pane.py -h

usage: pane.py [-h] -b BLACKLISTS -p PCAP

options:
  -h, --help            show this help message and exit
  -b BLACKLISTS, --blacklists BLACKLISTS
                        The name of file that contains the
blacklists
  -p PCAP, --pcap PCAP  The name of pcap file to analyze
```

Il pcap che noi abbiamo utilizzato è stato catturato con Wireshark

Costruzione del PCAP

All’interno del progetto c’è un software chiamato phammer.c che permette la creazione di pacchetti UDP senza il calcolo di Checksum. Questo software ci permette di creare pacchetti quindi con IP di nostra scelta e quindi anche indirizzi appartenenti ai file in blacklist. Può essere utilizzato per fare test di diverso tipo, controllando il funzionamento del programma e la sua classificazione degli indirizzi IP.
Possiamo passare a questo programma come argomento un indirizzo sorgente specifico, in questo modo verrà inviato un solo pacchetto derivante da tale indirizzo. In caso non venga specificato il programma invierà una serie di pacchetti con indirizzo sorgente randomico.
Eseguendo phammer -h

```
Usage: phammer [-h] -d <destination> [-a <source>] [-n <num>]
-h                  [Print help]
-d <destination>    [Address of receiver]
-a <source>         [Source address]
-n <num>            [Number of packets to send]
```

Nei nostri test nello specifico abbiamo mandato IP di gravità basso e medio più di una volta. Chiaramente se il programma reperisce che l’indirizzo che riceve è già in analisi, non lo

invierà nuovamente a IO. Questo permette di risparmiare tempo considerando i tempi di attesa del processo di analisi

Analisi del PCAP

Il programma principale pane.py analizza il PCAP passatogli tramite librerie python apposite. Scorre ogni messaggio e quindi sia gli IP sorgenti e destinazioni per analizzarli.

- Se l'indirizzo IP è locale, passa oltre poiché non può essere in blacklist
- Se l'indirizzo IP non è in blacklist, passa oltre
- Se l'indirizzo IP è presente in una delle blacklist istanziate all'avvio del programma, controlla che l'indirizzo non sia già passato e, se così non fosse, lo invia a IO e lo inserisce nella lista degli IP in blacklist già analizzati

Il controllo della Blacklist e l'utilizzo delle API sono gestiti da componenti esterno a pane.py

Blacklist

Per la gestione degli indirizzi IP in blacklist viene utilizzato un Patricia Tree, un apposito radix tree per il salvataggio di indirizzi IP. La libreria python si chiama Pytricia, la pagina github del progetto è la seguente https://github.com/jsommers/pytricia.
Una volta avviato il programma viene analizzato il file contenente la lista di blacklist per scaricarle, vengono controllate tutte e i vari indirizzi salvati nel Patricia Tree. Ogni volta che viene letto un indirizzo dal file pcap procediamo con il controllo all'interno della struttura per vedere se tale indirizzo potrebbe essere un buon candidato per il controllo tramite IntelOwl, se presente allora viene inizializzato un nuovo Job altrimenti passiamo oltre.

API

Le API di IO hanno una documentazione reperibile al seguente link: https://pyintelowl.readthedocs.io/en/latest/.
Il nostro uso nello specifico è presente nel file intel_sender.py che contiene i metodi necessari per comunicare con IO. Il suo ruolo è semplice:

- Invia un IP da analizzare salvando il numero del Job

```
client.send_observable_analysis_request(ip, analyzers_requested=["Pulsedive"],
observable_classification="ip")
```

- Interroga lo stato di esecuzione del Job

```
client.get_job_by_id(job_id)
```

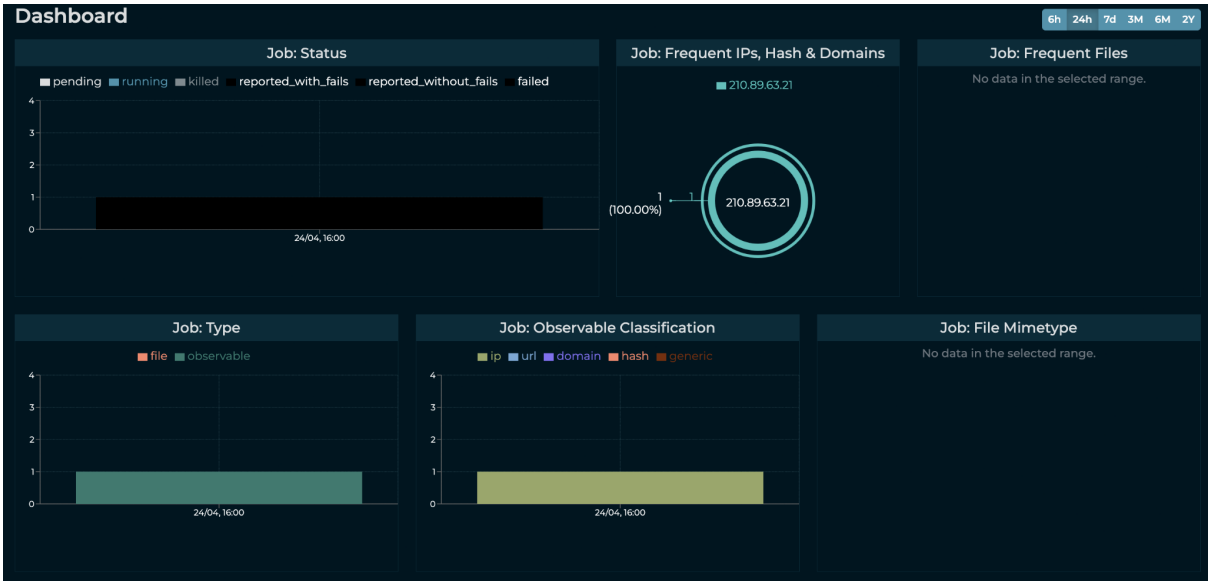
- Se completato può eliminare il Job se il rischio è low

```
client.delete_job_by_id(job_id)
job['status']=='reported_without_fails'
```

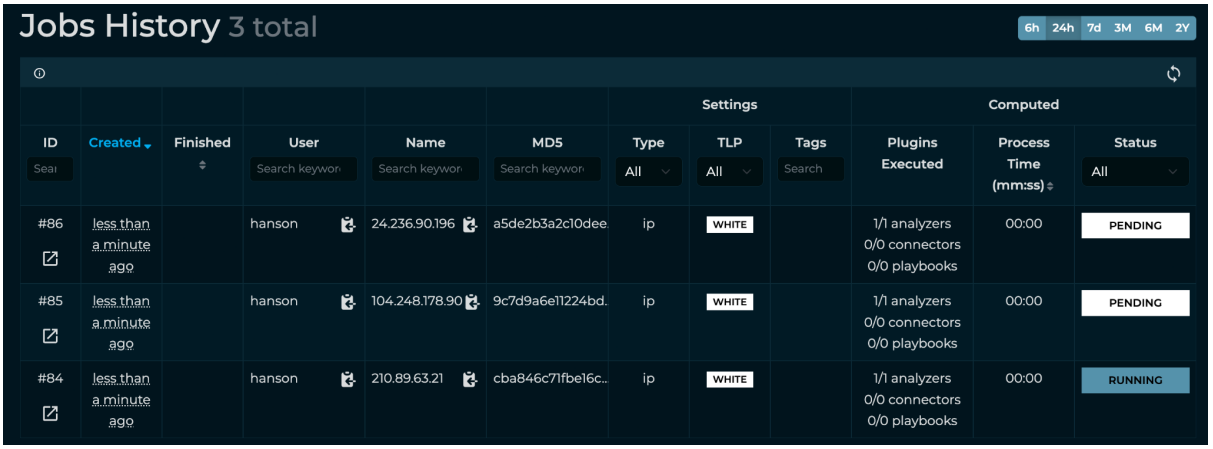
Per fare questo è necessario la chiave spiegata nei prerequisiti da inserire all'interno della configurazione

IntelOwl: interfaccia e funzionamento base

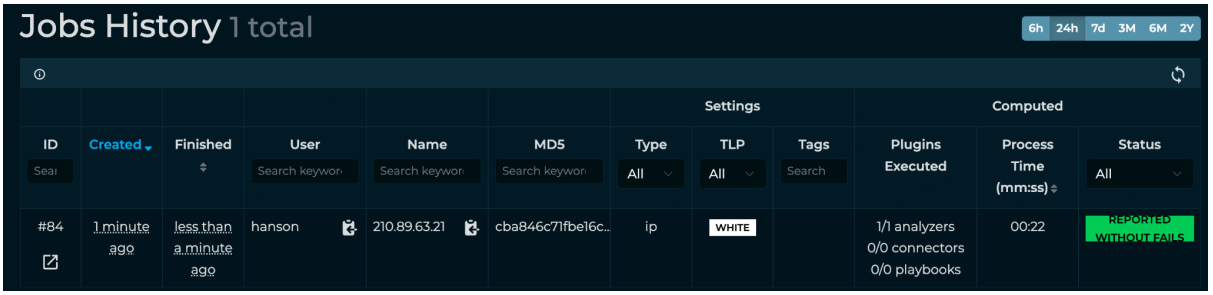
L’interfaccia principale di IO è quella mostrata qui sotto. Quello che verrà visualizzato sono solamente i task con un rischio medio o superiore dopo la prima scrematura iniziale. Questo è stato necessario poiché il processo di controllo di IO è relativamente lento anche con pochi controlli



I Jobs sono qui sottostanti. Il controllo di ogni IP corrisponde a un Job e possono essere inviati più di uno in contemporanea



I Job completati hanno un report come quello sottostante



IntelOwl offre la possibilità di utilizzare più analyzer per lo stesso observable. Per motivi di efficienza è stato scelto di utilizzarne solamente uno per svolgere le relative analisi, ovvero pulsedive.

Test

Per i nostri test abbiamo creato un apposito PCAP (fornito nella cartella) contenente degli IP malevoli da pacchetti creati con phammer. Gli IP in blacklist sono alcuni ma solamente 1 ovvero 210.89.63.21 viene considerato di rischio medio da parte di IO.
Il test quindi ci ha permesso di vedere che l'analisi degli IP di blacklist avviene una volta soltanto e che i test con rischio basso vengono successivamente eliminati. I passi si possono notare bene poiché il tempo di check in IO permette di vedere volta volta lo svolgimento dei Job e quindi il procedere degli IP e il loro smaltirsi