Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

Automazione della raccolta di evidenze per le analisi di un Incident Response tramite Ansible

Relatore:
Prof. Marco Prandini

Presentata da:
Alessandro Filippini

Correlatore:
Prof. Davide Berardi

Correlatore:
Niccolò Baldi

Sessione IV
Anno Accademico 2021/2022

Per gli appassionati,
per chi conosce la storia,
per chi la scoprirà ...

Indice

Elenco delle figure

Elenco dei frammenti di codice

Capitolo 1

Introduzione

Un Cyber Security Incident è un evento possibile o imminente che mette a rischio la riservatezza, l'integrità o la disponibilità delle informazioni o di un sistema informativo; oppure costituisce una violazione o una minaccia di violazione della legge, politiche o procedure di sicurezza [1].

La Digital Forensics and Incident Response (DFIR) è una branca dell'Information Security che si occupa di identificare, investigare, documentare e rimediare ad attacchi informatici o altre violazioni di sistemi/infrastrutture ad uso personale o aziendale. Può essere a sua volta divisa in due "sezioni"; la Digital Forensics, ramo delle scienze forensi dedicato alla ricerca e allo studio delle prove informatiche (digital evidence), che ha ad oggetto l'esame di file, dati, attività degli utenti, log di sistemi operativi e altre fonti digitali al fine di determinare se un attacco, evento o violazione siano state effettivamente perpetrate e chi potrebbe esserne l'autore; l'Incident Response, ossia l'insieme dei processi e attività che un'organizzazione dovrà seguire per rilevare e contenere una violazione dei propri dati/sistemi e per successivamente ripristinare la situazione pre-esistente l'attacco.

1.1 Situazione Odierna

Negli ultimi anni, una serie di eventi significativi ha eccessivamente velocizzato il processo di digitalizzazione delle infrastrutture pubbliche e private, non ancora sufficientemente mature in materia di Information Security o, talvolta, sprovviste dei tempi e fondi necessari per l'implementazione di sistemi complessi e sicuri. Per continuare ad operare in pieno periodo di pandemia ad esempio, numerose aziende ed istituti di istruzione hanno adottato mezzi di comunicazione digitali, spesso configurati in maniera frettolosa o con poca cura degli aspetti di security, creando l'opportunità per numerosi malintenzionati di intromettersi, ad esempio, in video-conferenze aziendali o addirittura governative. Naturalmente, laddove le piattaforme e gli strumenti digitali divengono critici, accresce anche l'interesse dei cyber criminali, che vedono in tali bersagli delle vere e proprie opportunità di business. Ormai sta scomparendo la figura dell'hacker solitario che compromette sistemi per diletto, in favore della diffusione di gruppi professionali di hacker che offrono le proprie competenze a fini, spesso, delinquenziali.

Dall'altro lato del campo, è ancora presente una carenza di professionisti [2] in grado di testare e mettere adeguatamente al sicuro sistemi e infrastrutture complesse, oltre a mancare ancora una concreta sensibilità e comprensione, da parte degli utenti, in materia di minacce o, più in generale, sicurezza informatica, vedi Figura 1.1. Per tentare di bilanciare questa carenza di risorse e competenze sono nate di recente delle iniziative per condividere gratuitamente informazioni e strumenti legati alla sicurezza informatica [3], come la piattaforma "No More Random" pubblicata da Europol.

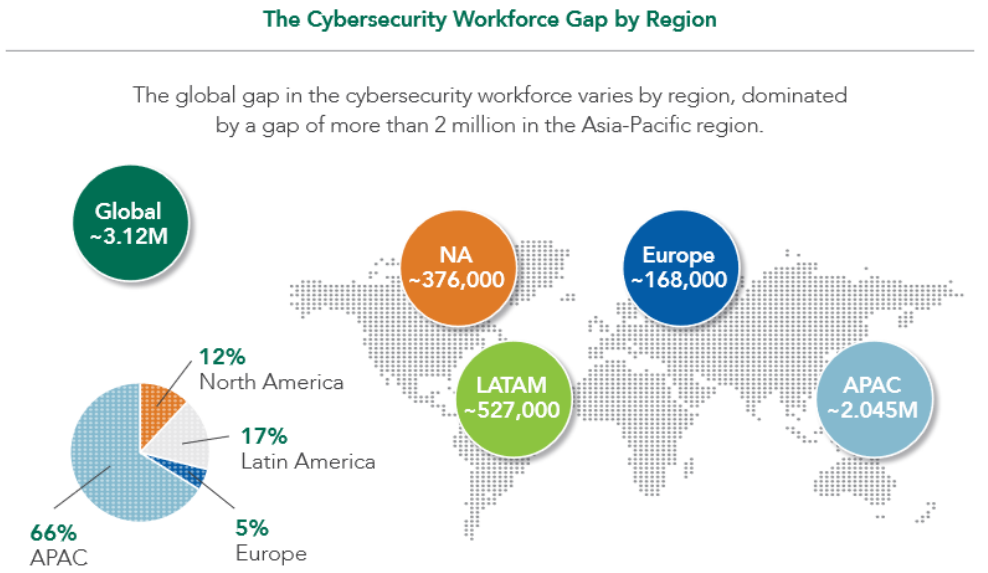

Figura 1.1: Forza lavoro mancante nel settore della sicurezza informatica

1.2 Scopo del progetto

Attualmente non esiste una soluzione tecnologica che permetta di automatizzare il processo di acquisizione dei dati/log da diversi sistemi potenzialmente compromessi, e al contempo di elaborare tutte le differenti fonti informative acquisite (log, cronologie, registri, ecc.), caricando i dati elaborati all'interno di un'unica piattaforma condivisa, dedicata alla successiva analisi da parte di più soggetti. In questa tesi verrà proposto un progetto che si pone l'obiettivo di colmare tale mancanza; non sarà interamente costruito dalle fondamenta, ma consisterà nel combinare differenti strumenti che svolgono già egregiamente il loro lavoro. L'intero sviluppo di un programma, in grado di svolgere molteplici mansioni complesse, richiederebbe uno sforzo considerevole, sia di risorse umane che finanziarie; conviene piuttosto combinare fra loro singoli strumenti specializzati [4], lasciando la responsabilità agli esperti della specifica materia di mantenerli affidabili nel tempo.

1.3 Struttura del documento

Nel Secondo Capitolo verranno posti gli obiettivi che vuole raggiungere il progetto. Il Terzo Capitolo andrà ad elencare i vari componenti del progetto, spiegando i vari motivi per cui stati scelti. Il Quarto Capitolo mostra l'effettiva implementazione degli strumenti e come sono stati gestiti. Infine, il Quinto e ultimo Capitolo conclude la tesi proponendo dei possibili miglioramenti.

Capitolo 2

Obiettivi

Prima di passare alla parte più concreta del progetto, è necessario fissare gli obiettivi da raggiungere. Essi sono fondamentali per scegliere adeguatamente i vari strumenti e la loro conseguente implementazione.

2.1 Ottimizzazione dei tempi

Ridurre i tempi necessari per completare un'attività di Incident Response, mantenendo comunque un alto livello di efficacia, è conveniente sia per l'analista, che nello stesso lasso di tempo potrà dedicarsi ad un numero maggiore di casi, sia per la vittima di un attacco informatico, che potrà ripristinare più velocemente i propri sistemi e servizi, riducendo in tal modo i significativi impatti economici che tali eventi comportano.

Considerato il gran numero di applicativi ad oggi utilizzati sulle macchine personali o sui sistemi server, e la mole di dati eterogenei dagli stessi generati, risulta estremamente complesso incrociare "a mano" le differenti fonti per costruire un'unica linea temporale di eventi da sottoporre all'analisi. Inoltre, ogni singola applicazione registra gli eventi con un proprio specifico linguaggio, pensato per essere facilmente compreso da determinati programmi. Il processo di acquisizione e interpretazione, in inglese parsing, dei predetti dati richiede un tempo elevato e risulta altamente suscettibile di errori umani

(dipendenti, ad esempio, da un errata configurazione del *tool* in uso in quel momento). Per risolvere tali problemi è necessario stabilire una procedura che possa essere automatizzata [5] e parallelizzabile, affinché il numero di macchine compromesse, da cui estrarre le necessarie fonti da analizzare, non divenga il "collo di bottiglia" dell'intera attività di analisi.

La comunicazione efficace fra i membri di un team è un altro punto fondamentale per una buona e rapida riuscita delle analisi di Incident Response [6]. Un'unica piattaforma per l'analisi condivisa si rivelerebbe dunque perfetta per questo scopo, permettendo a tutti i membri del team di essere aggiornati in tempo reale sugli sviluppi delle analisi svolte dai colleghi e di esaminare dati interpretati nello stesso formato. Inoltre, permetterebbe di eliminare lo spreco di tempo e risorse spesso derivante dalla ripetizione di determinate analisi svolte da diversi membri del team, che non essendo perfettamente aggiornati sul lavoro dei colleghi rischiano di ripetere un'analisi già svolta in precedenza.

2.2 Ottimizzazione dei costi

Anche l'equilibrio tra tempi e costi deve essere attentamente valutato e tenuto in considerazione: velocizzare un processo non sempre si traduce in una riduzione dei costi. Per esempio, l'adozione di un calcolatore più performante permette di ridurre i tempi di estrazione ed elaborazione delle evidenze, a fronte di costi maggiormente elevati da sostenere per hardware e risorse energetiche. Oltre a questo, va sempre tenuto conto del fattore prezzo, che se diviene competitivo mantenendo tempistiche ragionevoli, rende di certo il prodotto più vendibile sul mercato.

Uno dei fattori cruciali per valutare i costi che andranno sostenuti è la scelta delle tipologie di macchine e sistemi che andranno ad ospitare l'intero progetto. Se si intende dare priorità alla garanzia di riservatezza dei dati raccolti ed analizzati, allora con ogni probabilità la soluzione migliore risulterebbe essere un implementazione on-premise, con la quale si avrebbe

un completo controllo sull'accesso ai dati e programmi. Tuttavia, un'infrastruttura locale ha dei costi iniziali piuttosto elevati e richiede una periodica manutenzione degli ambienti fisici oltre a quelli software; considerati gli obiettivi di questo progetto, occorre trovare un compromesso e fare affidamento sulle garanzie di sicurezza fornite dai principali provider di Cloud computing. Un ulteriore punto a favore del Cloud, è inoltre la flessibilità con la quale si possono gestire le macchine e gli spazi di archiviazione, circostanza che permette di risparmiare notevoli costi nei periodi in cui le risorse dovessero rimanere inutilizzate. Nel contesto del presente progetto, il Cloud è senza ombra di dubbio la soluzione più conveniente, in quanto soltanto durante le attività di Incident Response verrà effettivamente utilizzata l'infrastruttura e i relativi sistemi e servizi. [7].

2.3 Riduzione competenze trasversali

L'evento chiave per la risoluzione di un analisi di Incident Response potrebbe nascondersi tra gli eventi registrati nei log di Windows, di un Firewall, di un filesystem oppure nella cronologia di navigazione di un Browser, tutte fonti aventi il proprio formato, percorso e sintassi: ogni log è progettato per essere interpretabile in un linguaggio a cavallo tra quello umano e quello macchina. È necessaria una notevole esperienza per analizzare efficientemente queste fonti così eterogenee fra loro, anche nei casi in cui è possibile accedere rapidamente a valida documentazione. Un singolo individuo con le competenze necessarie per le fasi di acquisizione, elaborazione/interpretazione ed analisi di tutte le differenti tipologie di eventi, fa parte di un gruppo piuttosto ristretto e non è facile da reperire sul mercato.

Come menzionato nel capitolo precedente, per ridurre le barriere di ingresso e rendere maggiormente accessibile la conoscenza della materia, c'è stato un impegno da parte di aziende ed enti governativi nel condividere gratuitamente informazioni e strumenti legati al mondo della Cyber Security e dell'Incident Response. Un analista alle prime armi farebbe meglio a svi-

luppare prima competenze verticali sulle tematiche di interesse, lasciando le competenze trasversali come secondarie o facoltative. Realizzare un team di specialisti è probabilmente più conveniente ed efficace che cercare un singolo *factotum* a cui affidare ogni fase del processo di Incident Response e l'analisi di ciascun tipo di evidenza.

Capitolo 3

Analisi Progettuale

Per scegliere gli strumenti necessari per l'analisi di Incidente Response, bisogna anzitutto comprendere quali funzioni dovranno svolgere. Inizialmente, i dati dovranno essere acquisiti per essere analizzati in un ambiente esterno rispetto ai dispositivi compromessi. Gli eventi acquisiti proverranno sicuramente da fonti diverse, dalle quali saranno estraibili log con campi eterogenei fra l'uno e l'altro, per cui il passaggio successivo riguarderà l'interpretazione (*parsing*) delle differenti fonti in una modalità che permetta di uniformare fra loro, in un formato di output condiviso, le differenti righe di log/eventi interpretati. Una volta preparati, i dati acquisiti ed interpretati potranno essere indicizzati all'interno di una piattaforma condivisa dedicata alle analisi.

Per i motivi citati in precedenza, il processo di acquisizione e preparazione dei dati da analizzare sarebbe molto vantaggioso se potesse essere automatizzato, lasciando la possibilità di monitorarne il flusso di esecuzione qualora fosse necessario risolvere imprevisti o adattarlo a casi particolari. Inoltre, tutti gli strumenti scelti dovranno essere supportati e regolarmente aggiornati, per non rimanere scoperti nel caso uscissero nuovi bug o vulnerabilità, ma soprattutto per assicurare di avere sempre uno strumento di in grado di interpretare le più recenti versioni delle fonti di interesse.

3.1 AWX - Ansible Tower

Ansible [8] è uno strumento da linea di comando che permette di automatizzare operazioni informatiche. Può essere utilizzato per la gestione di servizi, la configurazione di applicazioni, l'installazione di strumenti, l'aggiornamento di sistemi e tanto altro, tramite delle "ricette" contenenti i vari passaggi da seguire. Per semplificare la scrittura delle ricette, o *playbook*, i task vengono implementati utilizzando dei moduli come base, che astraggono la gestione di comandi o chiamate ad API. Ansible viene fornito con un ampia gamma di moduli pre-installati, con la possibilità di aggiungerne ulteriori creati dalla comunità.

Il punto forte di Ansible risiede nella possibilità di eseguire un playbook parallelamente su molteplici macchine, che vanno elencate all'interno di un inventario dal funzionamento simile a quello di un magazzino. All'interno di esso, ogni host avrà le proprie variabili e potrà essere assegnato a dei gruppi, che a loro volta conterranno ulteriori variabili, dando la possibilità di limitare l'esecuzione di determinati playbook ad alcuni specifici sottoinsiemi di host.

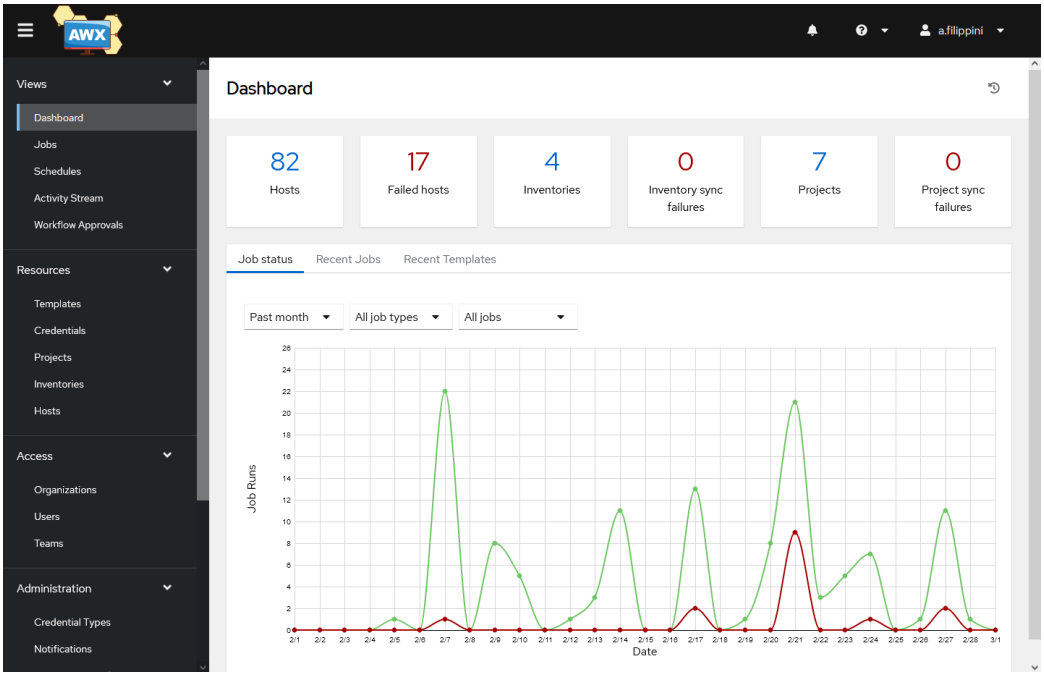

Figura 3.1: Dashboard di AWX

AWX [9] fornisce all'utente un'interfaccia web, REST API, e un *task engine* basato su Ansible. Mentre quest'ultimo è pensato per essere eseguito localmente su una macchina, AWX è uno strumento professionale pensato per essere utilizzato da più utenti in connessione remota. Aggiunge una gestione granulare dei permessi riguardo ai playbook, agli inventari e alle credenziali per accedere alle macchine, oltre a fornire una funzione che permette di creare un *workflow* per gestire l'esecuzione in sequenza di più playbook.

Integrare AWX in questo progetto contribuisce a eliminare la necessità di "competenze trasversali", uno degli obiettivi posti in precedenza, perchè permette una facile esecuzione dei playbook dedicati alle varie fasi dell'attività di Incident Response tramite una fruibile interfaccia grafica, che non costringe un utente a "manovrare" Ansible da linea di comando. Inoltre, risulta utile anche per chi ha il compito di sviluppare i diversi playbook, perché né registra internamente ogni singola esecuzione, includendo la durata e i valori di *output* di ciascun task eseguito, facilitando notevolmente la fase di

troubleshooting.

L'esecuzione dei playbook è delegata a dei "nodi" separati dall'host di AWX, e ciò permette di ridurne il numero o di limitare il lancio di un playbook ad un determinato sottogruppo di nodi. In uno scenario di Incident Response, in cui gli host compromessi da analizzare vengono spostati all'interno di una "bolla" isolata della rete, avere un nodo che possa fare da ponte tra tale VLAN dedicata ed AWX risulterà essenziale. Questa macchina verrà inoltre utilizzata nel progetto anche per altri scopi.

3.2 KAPE Kroll Artifact Parser and Extractor

Un altro degli obiettivi da perseguire per ridurre la durata di un'analisi di Incident Response è la creazione di un processo per l'acquisizione e interpretazione automatica degli eventi. Esistono svariati programmi dedicati a questa mansione, alcuni specifici per un particolare sistema operativo. Considerato che la quasi totalità degli incidenti informatici/violazioni avviene nel contesto di dispositivi Server o Client Microsoft Windows, questo progetto viene concentrato su tale sistema operativo, lasciando comunque la possibilità di estenderne l'utilizzo agli altri.

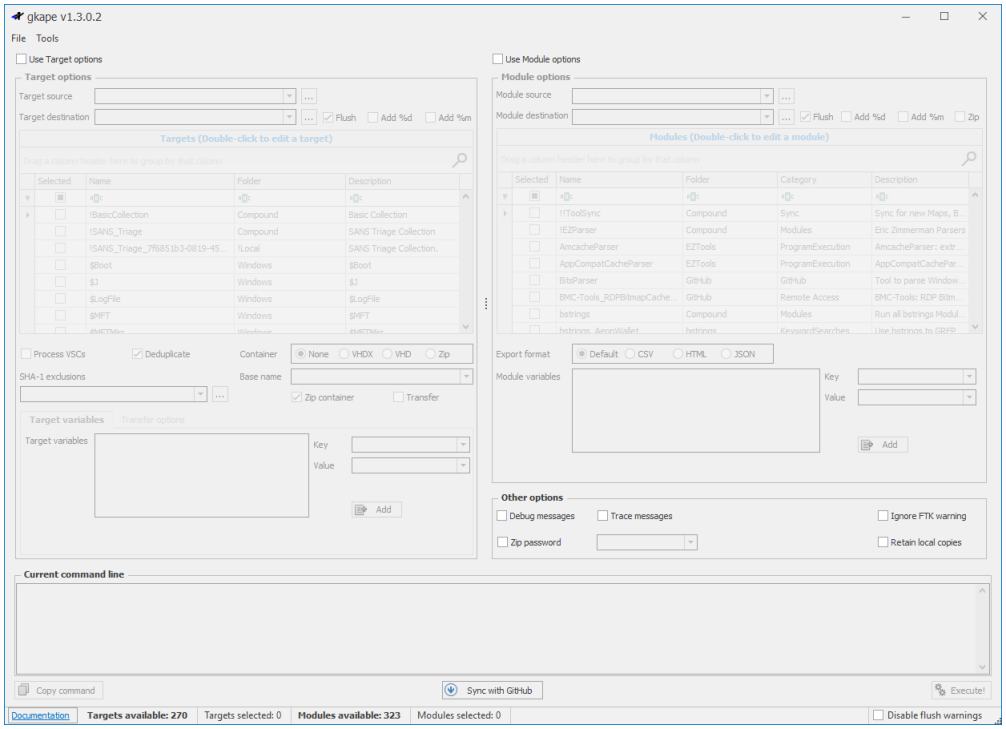

Figura 3.2: Interfaccia grafica per KAPE

KAPE [10] è un programma dedicato al Triage delle fonti di interesse di un sistema operativo; efficiente ed altamente configurabile, individua, estrae e interpreta le fonti di eventi utili dal punto di vista forense, ed è disponibile esclusivamente per i sistemi Microsoft Windows. Il suo ciclo operativo si divide in due fasi: la prima riguarda l'acquisizione dei registri o file di log del sistema operativo o di vari applicativi; la seconda (facoltativa nel nostro caso) riguarda il *parsing* dei dati acquisiti, che vengono esportati in diversi file CSV. KAPE è pensato per acquisire solo i log necessari legati ad un certo contesto o programma tramite una selezione dei "Target", una lista predefinita di path all'interno dei quali risiedono le fonti di interesse.

```
    Description: Chrome
    Author: Eric Zimmerman and Andrew Rathbun
    Version: 1.2
    Id: a56d0a8f-3229-489e-aea7-353d1f6f9639
    RecreateDirectories: true
    Targets:
        -
            Name: Chrome History
            Category: Communications
            Path:
↪   C:\Users\%user%\AppData\Local\Google\Chrome\User Data\*\
            FileMask: History*
```

Code 1: KAPE: Chrome Target

Una volta acquisiti i dati/fonti di interesse, KAPE può utilizzare dei "Moduli", composti da una lista di programmi da eseguire, che andranno ad arricchire, interpretare o filtrare i dati raccolti nella fase di Target, in alcuni casi aggiungendo funzionalità che KAPE non avrebbe (ad esempio l'acquisizione della memoria volatile tramite winpmem).

Il nucleo del programma risiede nell'eseguibile da linea di comando, ma esiste anche un'interfaccia grafica, illustrata nella Figura 3.2. Essa mostra tutte le opzioni disponibili: una volta scelte andranno a comporre, in un riquadro nella parte inferiore della finestra, l'anteprima dell'intero comando con cui verrà eseguito KAPE. La versione grafica può essere utilizzata per scopi di test, ma una volta stabiliti i Target e Moduli, KAPE verrà utilizzato esclusivamente da linea di comando.

3.3 Plaso

Come accennato in precedenza, ogni applicazione registra i propri eventi in un formato tipicamente proprietario; un browser, ad esempio, tipicamente salva la cronologia di navigazione, ricerche o download all'interno di un database, mentre un antivirus registra tipicamente tutti gli eventi di sicurezza rilevati, relativi a file malevoli o eseguibili acceduti dall'utente. Ciò considerato, sarà impossibile estrarre eventi generati da fonti diverse che vengano descritti tramite gli stessi campi o nello stesso formato. Oltretutto, in base al tipo di supporto su cui vengono salvati i log, il timestamp contenuto negli stessi potrebbe essere modificabile da fattori esterni, complicando la corretta creazione di una linea temporale attendibile. Serve dunque un programma in grado di riconoscere numerose tipologie di log differenti e di convertire il formato di ciascuna tipologia in uno standard condiviso, tale da permettere di aggregare tutti i log estratti all'interno di un unica coerente *timeline*.

Plaso è una raccolta di strumenti scritti in Python per la creazione di *supertimeline* e soddisfa pienamente i requisti posti in precedenza. Le supertimeline prodotte sono nativamente supportate dalla piattaforma Timesketch, che verrà descritta nella sezione successiva. La Supertimeline [11] è un'aggregazione di eventi, con il relativo timestamp, provenienti da numerose e differenti fonti. Torna utile incrociare varie sorgenti, nel caso in cui lo stesso evento sia stato registrato da diverse applicazioni, perchè si potrebbero contrastare tentativi di manipolazione della cronologia, detto in gergo *tampering*: per esempio se analizziamo nella stessa timeline eventi provenienti da un file presente su un disco, facilmente modificabile, e da una API in sola lettura.

3.4 TimeSketch

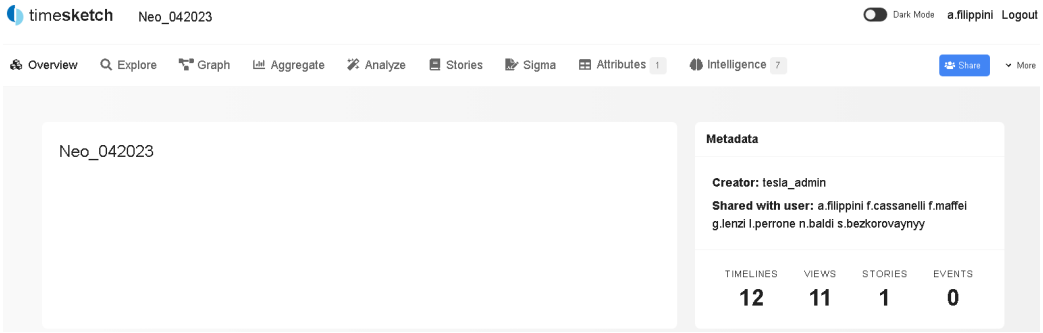

Figura 3.3: Landing Page di Timesketch

TimeSketch è una piattaforma open-source per l'analisi forense collaborativa di timeline, suddivise per ogni caso di Incident Response in uno sketch.Offre svariati strumenti per cercare e aggiungere note, commenti, tag ad eventi rilevanti individuati nel corso delle analisi.

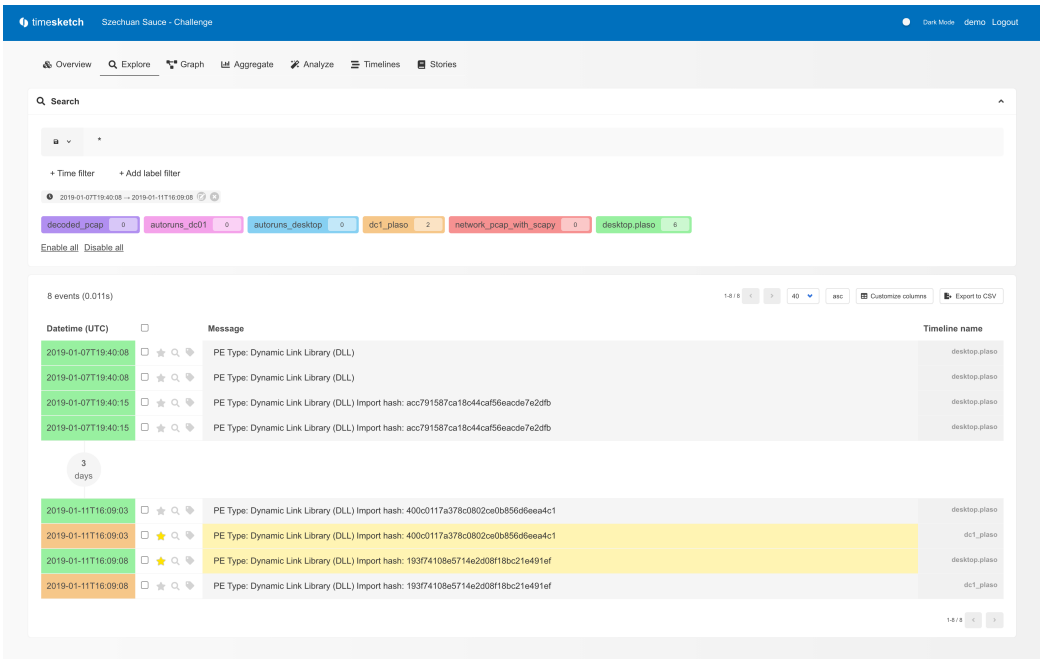

Figura 3.4: Ricerca incrociata su varie timeline in Timesketch

Rispetto ad altri strumenti, è capace di gestire rapidamente milioni di eventi, grazie al motore OpenSearch [12] basato su Elasticsearch. È possibile fare intelligence tramite Sigma e Yara Rules [13], automatizzando il processo di identificazione e aggiunta di tag ad IoC [14], pubblicamente noti o incontrati in passato. Partendo dagli eventi segnati come malevoli o dalle ricerche salvate dagli analisti, sarà possibile generare una bozza di report, contenente un anteprima degli eventi di interesse, grazie alla sezione *Stories*. TimeSketch è divenuto nel tempo uno strumento ampiamente diffuso e, per effetto di ciò, numerosi tool dedicati alla DFIR hanno integrato una funzione di export di log/eventi in un formato nativamente compatibile (*opensearch_ts*). Non si è dunque legati esclusivamente all'utilizzo di Plaso, per creare timeline analizzabili con la piattaforma TimeSketch, ma è possibile anche importare direttamente eventi o log da SIEM o network appliance.

Ma il principale punto di forza dello strumento TimeSketch resta la possibilità di lavorare contemporaneamente in più utenti, su un'unica piattaforma per Incident Response Analysis, sulla quale è possibile vedere in tempo reale i risultati delle analisi dei propri colleghi del team di IR, traendone i vantaggi discussi nella sezione 2.1. Unico "neo", tutta la gestione degli utenti e dei permessi deve essere svolta da linea di comando, ma rimane un compito che non serve venga assegnato agli analisti di IR in sè, per cui non presenta un problema nel contesto di questa tesi.

Capitolo 4

Implementazione

4.1 Preparazione

Nella presente sezione verranno approfondite le diverse scelte effettuate durante l'implementazione del progetto.

L'architettura, illustrata in Figura 4.1, mostra uno scenario in cui le macchine potenzialmente compromesse, oggetto di successiva analisi, vengono isolate dal resto della rete aziendale. Per evitare complicazioni legate alla mancanza di supporto o compatibilità, le ultime versioni di Windows testate con questo progetto saranno Microsoft Windows Server 2012R2 e Microsoft Windows 8.1, che andranno rispettivamente in EndOfLife il 10 Ottobre 2023 e il 10 Gennaio 2023. Oltre ai vari obiettivi definiti nel precedente capitolo, bisogna anche considerare che uno strumento dedicato alla DFIR dovrebbe "contaminare" il meno possibile le evidenze. Pertanto nelle scelte di implementazione sono stati adottati alcuni accorgimenti per limitare il "rumore" generato dai tool utilizzati, specialmente con KAPE ed Ansible, essendo gli unici strumenti che di fatto si interfacciano direttamente con le macchine compromesse.

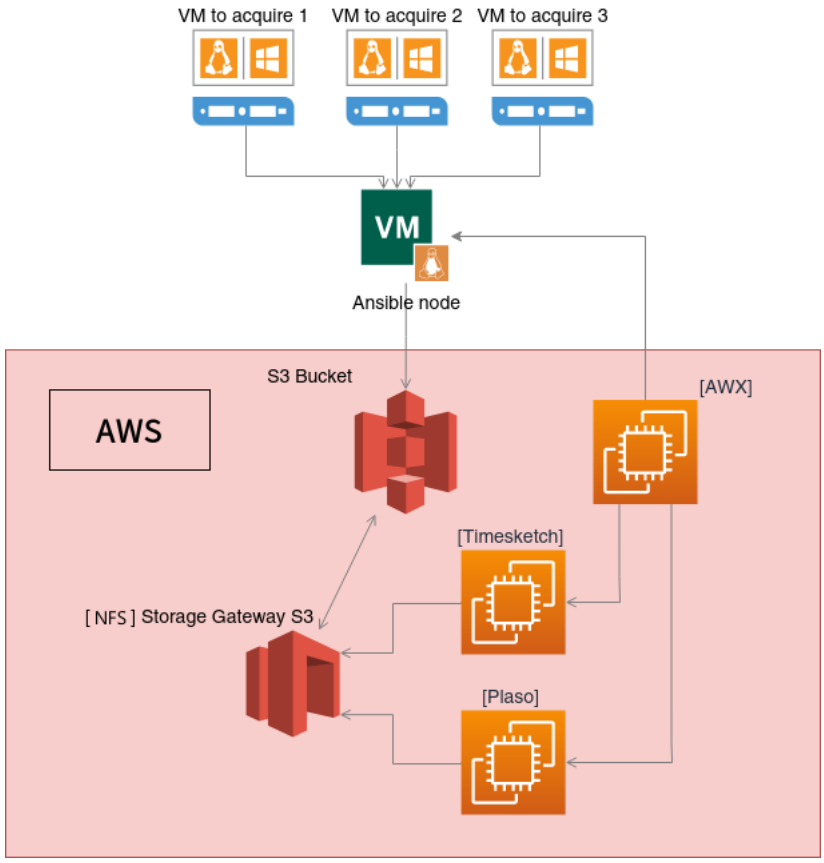

Figura 4.1: Architettura del progetto

4.1.1 AWX

La macchina che ospiterà AWX non richiede un particolare sistema operativo, visto che il metodo consigliato per installare la piattaforma [9] si basa sul sistema di container Docker.

Esistono due opzioni principali per l'aggiunta di un progetto ad AWX: archiviare i file all'interno di una cartella locale nella macchina AWX, oppure utilizzare una repository esterna. Data la natura di questo progetto, possibilmente soggetto a costanti aggiornamenti e test, la seconda opzione risulta quella maggiormente flessibile. Quando viene lanciato un playbook, si può scegliere di abilitare un'opzione per aggiornare automaticamente il progetto di cui fa parte. Al contrario dei playbook, gli inventari possono essere

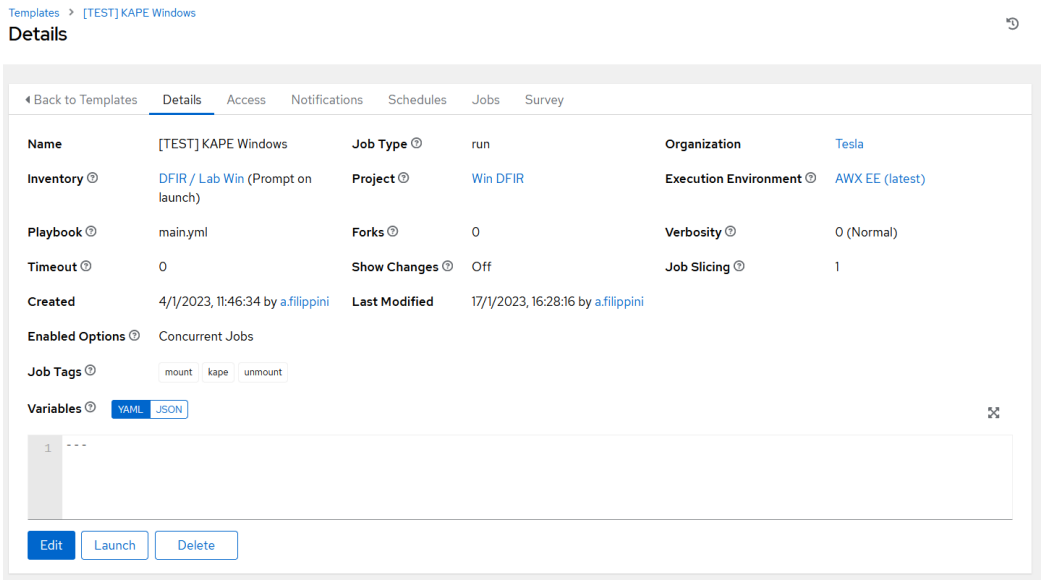

Figura 4.2: Template per il lancio di un playbook su AWX

manualmente popolati tramite l'interfaccia di AWX. Ciò tuttavia potrebbe creare incongruenze con le copie utilizzate per scopi di test, dunque anche essi dovranno verranno allineati utilizzando una repository Git.

AWX ha una sezione dedicata alla gestione delle credenziali, che possono essere utilizzate per autenticarsi a macchine remote oppure servizi API. Ad esempio, si può aggiungere un Access Token per eseguire il pull di una repository Git. La piattaforma può essere utilizzata da più utenti, ed offre una gestione granulare dei permessi riguardanti l'accesso a progetti, inventari e credenziali, ma ciò non rientra nel focus di questo progetto, che richiede una piattaforma condivisa per le attività di analisi piuttosto che per la fase di acquisizione dei dati.

4.1.2 KAPE

Nella descrizione precedente di KAPE abbiamo già visto che per creare un elenco di dati da acquisire occorre selezionare un inseme di Target. Nel caso uno dei file di interesse non risulti presente nel file system della macchina oggetto di acquisizioine, KAPE semplicemente lo ignorerà; quin-

di, teoricamente, si potrebbero utilizzare tutti i Target disponibili, andando tuttavia ad allungare la durata dell'acquisizione e le dimensioni dell'archivio, che sicuramente risulterebbe poi contenere dati potenzialmente inutili per le analisi. Serve dunque una via di mezzo, ovvero una selezione di Target che possano coprire un sufficiente perimetro di fonti di interesse, tra Browser, log e registri del sistema operativo.

La sezione Moduli non presenta nessuno strumento utile per arricchire le informazioni raccolte, dunque può essere nel nostro caso ignorata.

KAPE cerca i dati da acquisire all'interno di un path specificato dall'utente. Nel caso in cui non risulti possibile effettuare un'acquisizione statica (offline), KAPE dovrà per forza essere eseguito dalla macchina compromessa. L'output dell'acquisizione può essere compresso all'interno di un archivio ZIP e nonostante, la sua dimensione relativamente ridotta, non è scontato avere a disposizione spazio libero sufficiente sul disco della macchina target. Inoltre, il filesystem NTFS, predefinito da Windows XP in poi, registrerebbe il rumore generato dall'estrazione dell'archivio contenente KAPE nel registro $LogFile [15, 16].

Per risolvere entrambi i problemi, KAPE potrebbe essere allocato all'interno di un disco da poter montare su ogni macchina. L'uso di un disco tradizionale viene escluso, perchè l'aggiunta richiederebbe un intervento manuale su ogni singola macchina. Andando per esclusione, rimane solo l'opzione di utilizzare un disco di rete, la cui gestione può essere inoltre automatizzata. Per evitare problemi di compatibilità con alcune versioni Windows, specialmente le più datate, la cartella contenente KAPE verrà condivisa tramite il protocollo Samba. Il disco di rete avrà anche il ruolo di destinazione per KAPE, quindi è sempre bene verificare che ci sia sufficiente spazio libero a disposizione, o predisporre un molto capiente per essere certi di non avere problemi. Avendo il resto dell'architettura su AWS, lo spazio di storage verrà ospitato su un bucket di S3; un grande vantaggio di utilizzare S3 sarà l'assenza di un limite prefissato, che toglie il pensiero di dover prevedere quanto andrebbe ad occupare un'acquisizione.

Per assicurare l'omogeneità dei risultati di KAPE, l'installazione viene "congelata" in un immagine ISO; purtroppo molti file Target superano il limite imposto dallo standard ISO9660 [17], che definisce la lunghezza massima di un filename come 8 caratteri per il nome e 3 per l'estensione. Esistono degli standard aggiuntivi che possono aggirare questo problema, come Joliet che supporta nomi a 64 caratteri Unicode.

Dopo vari tentativi di creare una immagine ISO con Joliet, Rock o Ibrida, l'unica combinazione risultata funzionante è ISO9660 conformance level 3 con filesystem UDF. I sistemi Windows 95, 98 e ME non sono compatibili con UDF, ma è poco rilevante dato che ci siamo prefissati come limite i sistemi Microsoft Windows 8.1 e Microsoft Windows Server 2012R2.

Il formato di uscita per le estrazioni di KAPE è tipicamente un semplice archivio ZIP, che rende gli spostamenti attraverso la rete più veloci rispetto alla copia sequenziale di piccoli file oltre a preservare i metadati originali. KAPE da comunque la possibilità di esportare in formato VHD o in una directory non compressa.

4.1.3 Plaso

Per evitare di sovraccaricare troppo la macchina che ospita TimeSketch, Plaso verrà eseguito su una macchina dedicata, che potrà essere spenta una volta terminate tutte le estrazioni. Anche in questo caso, Plaso viene distribuito in vari formati e pacchetti, tra cui un immagine per Docker. Nell'architettura mostrata in 4.1, Plaso accede ai dati nel bucket tramite Storage Gateway, che può condividere il bucket tramite Samba oppure NFS. Una volta terminata l'elaborazione dei dati acquisiti, la timeline generata sul bucket s3 sarà pronta per essere caricata su TimeSketch.

4.1.4 TimeSketch

Anche in questo caso, il metodo più veloce risulta l'installazione tramite i container Docker. Dopo aver seguito la guida ufficiale [18], gli utenti che

andranno ad accedere alla piattaforma, dovranno essere aggiunti tramite un strumento da linea di comando [19]. Anche le timeline possono essere caricate da linea di comando tramite un tool di importazione, ma richiede sempre uno sketch a cui essere associata, per cui all'apertura di ogni Incident Response ne verrà creato uno.

4.1.5 Ansible Node

AWX e il bucket S3 non saranno accessibili dalla rete isolata, per cui servirà un "ponte" che li colleghi all'esterno. AWX può essere scalato aggiungendo "Execution Node" e limitando il lancio dei playbook ad alcuni di essi. Attualmente, l'installazione del nodo è supportato solo dalle distribuzioni Linux della famiglia Red Hat [20]. L'altro ruolo di questo nodo sarà la condivisione del bucket S3, che ospiterà sia l'immagine di KAPE sia i dati estratti dalle macchine, per i motivi citati nella sezione 4.1.2.

4.1.6 Macchine da Acquisire

Ansible è pensato per gestire principalmente sistemi Linux, quindi predilige le comunicazioni tramite SSH. Anche se nelle versioni più recenti di Windows è stata semplificata la procedura di installazione di un server SSH, viene comunque consigliato l'utilizzo di WinRM, che è supportato da tutte le versioni utilizzate in questo progetto. Ansible mette a disposizione sulla propria repository [21], uno script per configurare automaticamente WinRM che andrà ad abilitare il servizio e configurare i certificati e i vari metodi per l'autenticazione. Quale metodo utilizzare dipenderà da come sono stati configurati i sistemi target prima di essere compromessi; ad esempio, se facessero parte di un dominio si potrebbe optare per Kerberos, oppure, qualora fossero disponibili delle credenziali di un utente amministratore locale, il metodo migliore sarebbe Basic. Il resto delle opzioni richiedono una configurazione aggiuntiva troppo macchinosa per lo scopo di questo progetto.

Questa configurazione "inquina" necessariamente la macchina oggetto di analisi, dato che viene abilitato un servizio per l'accesso remoto ed un'utenza amministrativa locale. Una volta terminate le analisi, è buona pratica ripristinare le macchine ad un backup antecedente all'intromissione e le analisi; nel caso in cui ciò non avvenga, si possono sempre annullare le configurazione applicate a WinRM e agli utenti locali.

4.2 Estrazione

Una volta preparata la "bolla" e configurati gli host target, le macchine sono pronte per essere acquisite ed analizzate. In questa sezione verranno descritti brevemente i vari passaggi che eseguirà il playbook, dall'estrazione fino alla timeline su TimeSketch.

La prima attività svolta dal playbook è quella di rendere accessibile il bucket S3 all'interno della macchina. KAPE non supporta come percorsi di destinazione degli indirizzi di rete, per cui la cartella condivisa dall'Ansible Node dovrà essere montata come disco. L'operazione richiede che il metodo di autenticazione scelto supporti la delega delle credenziali, altrimenti il disco verrà montato esclusivamente per la durata del singolo task. In alternativa, si può eseguire il task come utente SYSTEM. Lo stesso discorso vale per l'immagine ISO di KAPE.

La destinazione per le acquisizioni deve essere creata seguendo la struttura illustrata nella Figura 4.3. All'interno di "Acquisitions", ogni Incident Response ha la propria cartella, contenente a sua volta una cartella dedicata ad ogni macchina acquisita.

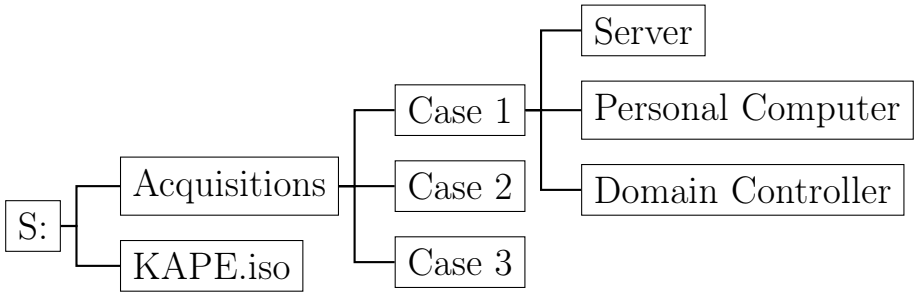

Figura 4.3: Struttura directory del bucket

Seguendo l'esempio nel Codice 2, Ansible andrà ad eseguire KAPE. Nel caso in cui venga rieseguita l'estrazione, l'archivio precedente non verrà sovrascritto perchè contiene nel nome il timestamp di inizio esecuzione. Terminata l'acquisizione, il playbook termina smontando i dischi montati in precedenza, per ripristinare il sistema allo stato precedente alle analisi.

```
1 .\kape.exe --tsource C: --tdest
  ↪ S:\acquisitions\<case>\<data>-<host> --zip <host> --target
  ↪ KapeTriage,!BasicCollection,!SANS_Triage,Antivirus,
  ↪ RecycleBin,$LogFile
```

Code 2: Esempio di un'esecuzione KAPE

Appena l'archivio sarà disponibile sul bucket S3, verrà lanciato un secondo playbook dedicato alla fase di creazione della timeline. Plaso offre la possibilità di analizzare un archivio senza doverlo estrarre, a fronte tuttavia di un tempo maggiore richiesto per l'elaborazione; dai test effettuati, risulta maggiormente conveniente estrarre l'archivio in una cartella temporanea.

La timeline generata verrà aggiunta allo sketch in cui vengono indicizzati tutti i dati dell'Incident Response, assegnadole il nome della macchina acquisita. Lo strumento di importazione da CLI per TimeSketch, prima di terminare il proprio processo, attenderà che la timeline sia stata effettivamente indicizzata, per una durata che dipenderà dalla quantità e dalla qualità

degli eventi presenti nei log. Questo comportamento predefinito potrà essere disabilitato, per ridurre il tempo complessivo in cui la macchina Plaso dovrà rimanere accesa, a scapito della possibilità di monitorare l'intero processo che, in tal caso, non terminerà quando i dati saranno effettivamente pronti per essere analizzati.

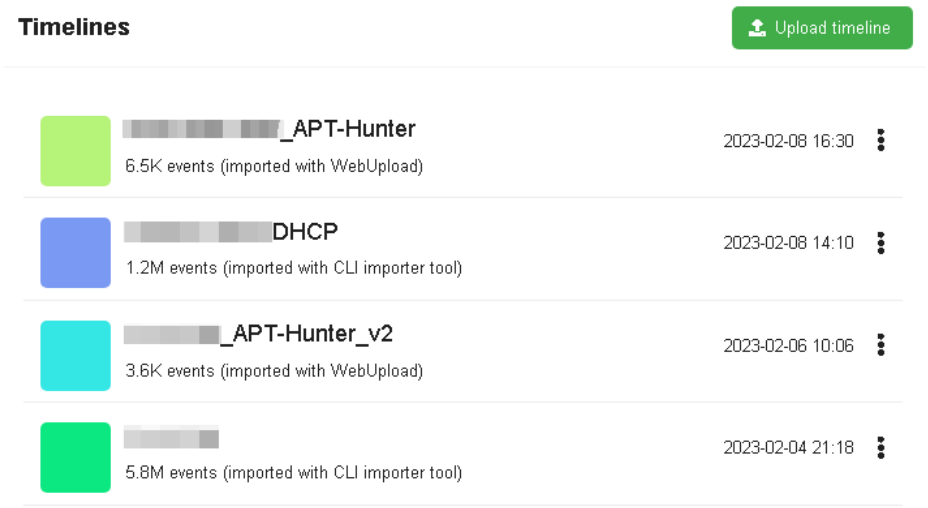

Figura 4.4: Elenco delle timeline disponibili

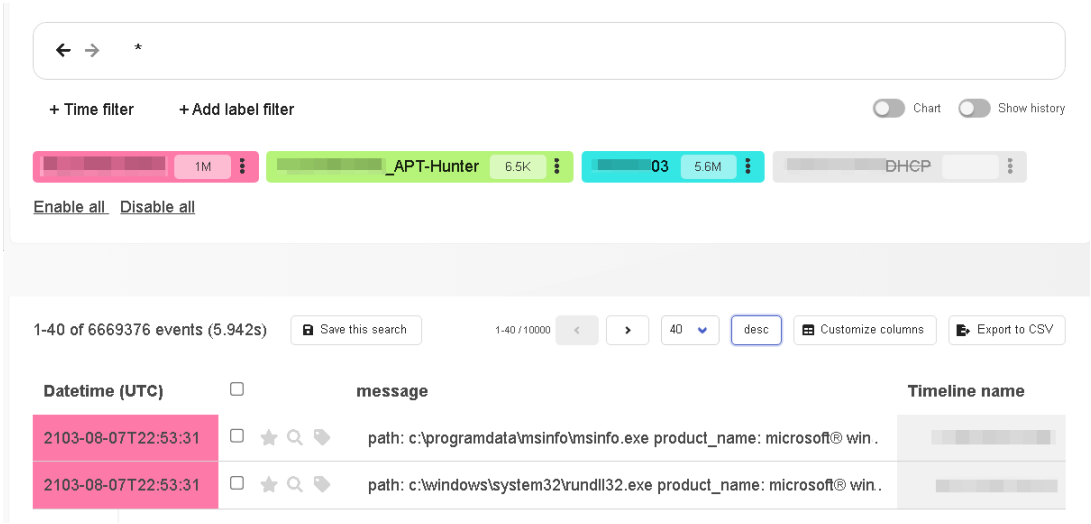

Figura 4.5: Analisi delle timeline caricate

Capitolo 5

Conclusioni

Negli scorsi capitoli sono state approfondite tutte le scelte riguardanti la creazione del progetto, l'architettura scelta, e le modalità con le quali possono comunicare diverse componenti, raggiungendo gli obiettivi posti nel Secondo Capitolo.

La piattaforma AWX permette la gestione automatica dell'acquisizione di evidenze tramite KAPE, ottenendo dagli host potenzialmente compromessi quanto necessario per un analisi di Incident Response. Viene poi sfruttato Plaso per uniformare ed interpretare le evidenze acquisite, eseguito sempre in maniera automatica tramite playbook di Ansible. Plaso produce una super-timeline che viene successivamente caricata ed indicizzata sulla piattaforma di analisi condivisa TimeSketch. Per ottenere questo risultato, è richiesto un unico analista che si occupi di eseguire i playbook nell'ambiente predisposto, mentre il resto del team deve solo attendere che i dati siano pronti per essere analizzati.

5.1 Sviluppi Futuri

Per mancanza di tempo, molte idee sono state scartate in questa fase, ma saranno implementate in futuro per migliorare la gestione e automatizzare ulteriormente alcuni processi. Si è data una maggiore importanza alla

creazione di una solida struttura modulare, che permetta miglioramenti o estensioni (ad esempio, integrando uno strumento di Triage per i sistemi operativi UNIX).

5.1.1 Indipendenza di Plaso

Per la natura stessa di Ansible, viene parallelizzata l'esecuzione dei singoli task, non di un intero playbook. Pertanto, prima di poter procedere al *parsing* dei dati con Plaso, Ansible deve attendere il termine di tutte le acquisizioni che si stanno eseguendo con KAPE.

Per rendere indipendente l'esecuzione di Plaso da KAPE, si potrebbero in futuro sfruttare alcuni strumenti offerti dallo stesso AWS, che permettono il lancio automatico di container se vengono verificate determinate condizioni.

5.1.2 Triage su altri Sistemi

Come detto in precedenza, KAPE è disponibile esclusivamente per i sistemi operativi Microsoft Windows. Sarebbe interessante utilizzare una soluzione analoga per il Triage su sistemi UNIX, da integrare con un apposito playbook.

All'interno di un'infrastruttura complessa sono presenti anche ulteriori strumenti per monitorare la sicurezza delle rete locale, come sistemi di Intrusion Detection, Firewall o SIEM; sarebbe interessante sviluppare un automatismo per l'acquisizione e indicizzazione dei log di tali sistemi all'interno di TimeSketch, considerando che nativamente non hanno un formato immediatamente comprensibile da TimeSketch e non risultano inoltre acquisibili con KAPE.

5.2 Osservazioni finali

La parte più delicata è stata la definizione di una configurazione per la rete "bolla", che deve essere implementata all'interno dell'infrastruttura di

un azienda terza, rimasta vittima dell'attacco informatico. Il progetto è stato pensato per poter predisporre assieme al cliente l'ambiente necessario, con un ridotto numero di passaggi da svolgere a cura di una risorsa dedicata, non necessariamente parte del team di Incident Response.

Ho avuto la fortuna di sviluppare questo progetto affiancato da alcuni esperti di Cyber Security, Digital Forensics e Incident Response, operanti nel settore da svariati anni. Senza la loro guida, la progettazione si sarebbe dilungata per mesi, e avrei probabilmente perso molto tempo a individuare e testare quali programmi avrebbero potuto soddisfare al meglio i requisiti e conseguire gli obiettivi iniziali.

Ho potuto testare questo progetto presso l'azienda per la quale attualmente lavoro, durante reali scenari di Incident Response; sicuramente questo fattore mi ha aiutato molto a comprendere, da vicino, quali fossero le priorità e le problematiche legate ad un concreto contesto di analisi di DFIR.

Bibliografia

[1] Ron Ross et al. «Protecting controlled unclassified information in nonfederal systems and organizations». In: *Protecting Controlled Unclassified Information in Nonfederal Systems and Organizations* (feb. 2020). DOI: 10.6028/nist.sp.800-171r2. URL: https://csrc.nist.gov/publications/detail/sp/800-171/rev-2/final.

[2] Emilio Granado Franco et al. Gen. 2022. URL: https://www.weforum.org/reports/global-risks-report-2022/in-full.

[3] Paul Mee e Chaitra Chandrasekhar. *Cybersecurity is too big for governments or firms to handle alone*. Mag. 2021. URL: https://www.weforum.org/agenda/2021/05/cybersecurity-governments-business/.

[4] P.H. Salus. *A Quarter Century of UNIX*. Addison-Wesley UNIX and Open Systems Series. Addison-Wesley Publishing Company, 1994. ISBN: 9780201547771.

[5] Timothy Grance et al. *Guide to Integrating Forensic Techniques into Incident Response*. en. Set. 2006. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=50875.

[6] Paul Cichonski et al. In: *Computer Security Incident Handling Guide* SP 800-61.2 (ago. 2012). DOI: 10.6028/nist.sp.800-61r2. URL: https://doi.org/10.6028/NIST.SP.800-61r2.

[7] Suchitra Landgey e D Wasiha Tasneem. «AWS Cloud Infrastructure vs Traditional On-Premise». In: *International Journal of Research Publication and Reviews* 2 (dic. 2021). DOI: 10.5281/zenodo.5761173. URL: https://doi.org/10.5281/zenodo.5761173.

[8] Red Hat. *Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy and maintain. Automate everything from code deployment to network configuration to cloud management, in a language that approaches plain English, using SSH, with no agents to install on remote systems.* 2012. URL: https://github.com/ansible/ansible.

[9] Red Hat. *AWX provides a web-based user interface, REST API, and task engine built on top of Ansible. it is one of the upstream projects for Red Hat Ansible Automation Platform.* 2016. URL: https://github.com/ansible/awx.

[10] Eric Zimmerman. *Introducing KAPE – Kroll Artifact Parser and Extractor.* Feb. 2019. URL: https://www.kroll.com/en/insights/publications/cyber/kroll-artifact-parser-extractor-kape.

[11] Kristinn Guðjónsson. *Mastering the Super Timeline With log2timeline.* Ago. 2010. URL: https://www.sans.org/white-papers/33438.

[12] Amazon. *What is OpenSearch?* 2021. URL: https://aws.amazon.com/what-is/opensearch/.

[13] Adam Lockett. «Assessing the Effectiveness of YARA Rules for Signature-Based Malware Detection and Classification». In: *CoRR* abs/2111.13910 (2021). DOI: https://doi.org/10.48550/arXiv.2111.13910. arXiv: 2111.13910. URL: https://arxiv.org/abs/2111.13910.

[14] Gerardo Canfora et al. «About the Robustness and Looseness of Yara Rules». In: *Testing Software and Systems.* A cura di Valentina Casola, Alessandra De Benedictis e Massimiliano Rak. Cham: Springer International Publishing, 2020, pp. 104–120. ISBN: 978-3-030-64881-7.

[15] Junghoon Oh, Sangjin Lee e Hyunuk Hwang. «NTFS Data Tracker: Tracking file data history based on $LogFile». In: *Forensic Science International: Digital Investigation* 39 (2021), p. 301309. ISSN: 2666-2817. DOI: https://doi.org/10.1016/j.fsidi.2021.301309. URL: https://www.sciencedirect.com/science/article/pii/S2666281721002341.

[16] Muhammad Sharjeel Zareen e Baber Aslam. «$Logfile of NTFS: A blueprint of activities». In: *17th IEEE International Multi Topic Conference 2014* (apr. 2015). DOI: 10.1109/inmic.2014.7097356.

[17] Volume ECMA. «File Structure of CDROM for Information Interchange». In: *Genf, Reprinted* 11.12.1987 ().

[18] *Install timesketch.* URL: https://timesketch.org/guides/admin/install/.

[19] *Admin CLI.* URL: https://timesketch.org/guides/admin/admin-cli/.

[20] Ansible. *Adding execution nodes to AWX.* URL: https://github.com/ansible/awx/blob/devel/docs/execution_nodes.md.

[21] Trond Hindenes. *Configure a Windows host for remote management with Ansible.* URL: https://github.com/ansible/ansible/blob/devel/examples/scripts/ConfigureRemotingForAnsible.ps1.

Ringraziamenti

Prima di tutti, vorrei ringraziare i professori Marco Prandini e Davide Berardi, che si sono resi disponibili e interessati per la scrittura di questa tesi.

Ringrazio anche tutti i miei collegi presso Tesla Consulting, in particolare Niccolò Baldi, farò tesoro di tutte le esperienze vissute insieme. Mi ritengo molto fortunato a poter collaborare con vari esperti nel settore della Sicurezza Informatica.

Ringrazio particolarmente i miei genitori, le mie sorelle e il resto della mia famiglia, non ho mai sentito il peso delle aspettative e sono sempre stati presenti in caso di bisogno.

Uno scontato ma necessario grazie va a tutte le amicizie che ho potuto coltivare negli anni, non ci sono parole per descrivere quanto mi abbiate aiutato dentro e fuori le aule. In ordine sparso, grazie particolare a: Paga, Evan, Suppa, Mago, Berto, Donno, Giaco, Juri, Alice, Angelo, Letizia ed Adriano.

Ringrazio infine Kai, che più di tutti mi ha dovuto sopportare negli ultimi due anni e in cui ho potuto trovare una seconda casa.