ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA

Dipartimento di Informatica – Scienza e Ingegneria

Corso di Laurea Magistrale in Ingegneria Informatica

REALIZZAZIONE E ANALISI DI UNA BASELINE PER IL RILEVAMENTO DI ANOMALIE IN APPLICAZIONI WEB

TESI DI LAUREA

in

Tecnologie e sistemi per la gestione di basi di dati e big data M

CANDIDATO

Mattia Pranzini
Matricola: 0000979223

RELATORE:

Prof. Paolo Ciaccia

CORRELATORE:

Dott. Angelo Neri

Anno Accademico 2021/2022

Sommario

1 Introduzione

Negli ultimi anni si sono verificati significativi sviluppi tecnologici che hanno coinvolto principalmente il settore informatico. In questo settore ci si trova spesso a dover fare i conti con vari tentativi di attacco, alcuni dei quali possono avere conseguenze catastrofiche.

La comunità informatica ha il compito di implementare gli strumenti per prevenire il verificarsi di un attacco, ma, qualora la prevenzione non risulti sufficiente, è necessario sviluppare un piano di ripristino una volta che un attacco abbia avuto successo. Sia gli utenti finali che gli operatori del settore, come amministratori di sistema e programmatori, devono prestare molta attenzione alla sicurezza IT. Chi è coinvolto nello sviluppo di applicazioni o nell'amministrazione dei sistemi deve essere costantemente aggiornato e formato per essere a conoscenza delle nuove tipologie di attacco sempre più diffuse.

Secondo un rapporto di marzo 2022 di Clusit, gli attacchi globali nel 2021 sono aumentati del 10% rispetto all'anno precedente e la loro entità risulta essere sempre più importante. Nel 2021, infatti, circa il 79% degli attacchi rilevati ha avuto un impatto significativo, l'anno prima erano solo il 50%.

A differenza del passato, queste operazioni sono state effettuate contro obiettivi ben definiti, come i settori governativo o militare, rappresentando il 15% del totale degli attacchi.

Nell'attuale ambiente digitale connesso, i criminali informatici utilizzano strumenti sofisticati per lanciare attacchi informatici alle aziende. I loro obiettivi includono PC, reti di computer, infrastrutture IT e sistemi informatici. Il tutto allo scopo di entrare in possesso di informazioni riservate o arrecare disagi all'organizzazione. In alcuni casi, i criminali richiedono anche ingenti riscatti, spesso in criptovaluta, per consentire al gruppo di riottenere la proprietà delle sue informazioni.

Questa tesi è stata svolta come completamento di un progetto di tirocinio presso il Cineca.

Il Cineca è un Consorzio Interuniversitario senza scopo di lucro formato da 102 Enti pubblici**:**

- 2 Ministeri
- 69 Università italiane,
- 31 Istituzioni pubbliche Nazionali (11 Enti di Ricerca, 7 Aziende Ospedaliere Universitarie-IRRCS, 11 Istituzioni AFAM, 1 Agenzia, 1 Parco Archeologico).

Fondato nel 1969 (come consorzio interuniversitario per il calcolo automatico nell'Italia nord-orientale), il Cineca è oggi il più grande centro di calcolo in Italia e uno dei più importanti al mondo. Operando sotto il controllo del Dipartimento di Ateneo e Ricerca, Cineca supporta le attività della comunità scientifica attraverso il supercalcolo e le sue applicazioni, realizza sistemi gestionali per l'Amministrazione Universitaria e il MUR, progetta e sviluppa sistemi informativi per la Pubblica Amministrazione, la sanità e le imprese.

Il Cineca è un punto di riferimento sempre più unico per l'innovazione tecnologica in Italia, con sedi a Bologna, Milano, Roma, Napoli, Chieti e oltre 900 dipendenti al servizio del sistema accademico e della ricerca nazionale.

Il Cineca, essendo un fornitore di servizi informatici in rete, è un ente potenzialmente soggetto ad attacchi informatici e, come strumento di analisi dei log, ha scelto di utilizzare il software IBM Security QRadar SIEM fornito da IBM.

Questo lavoro di tesi, si è concentrato sulla rilevazione degli attacchi che possano esaurire le risorse applicative e causare indisponibilità del servizio, oltre che sulla generazione di avvisi nel caso in cui alcuni tentativi abbiano avuto successo.

2 Tipologie di attacco

Un attacco informatico è un tentativo doloso e deliberato da parte di un individuo o di un'organizzazione avente lo scopo di danneggiare i sistemi informativi di un altro individuo o azienda. Gli aggressori cercano di sfruttare le vulnerabilità soprattutto nei sistemi aziendali ed è preoccupante sapere che i tassi di criminalità informatica aumentano ogni anno. Spesso, in seguito al successo di un attacco, i malintenzionati sfruttano la debolezza dell'organizzazione per chiedere un riscatto e promettere il recupero e il ripristino dei dati. Promesse in molti casi non mantenute. In particolare, il 53% degli attacchi informatici ha causato danni per oltre 500.000 dollari.

Le minacce informatiche non si concentrano solo su fattori economici, ma possono sorgere anche per altri motivi, come l'attivismo digitale, in cui sistemi e dati vengono compromessi per attuare una forma di mobilitazione di massa per indurre cambiamenti sociali e politici.

Alcuni dei tipi più comuni di attacchi sono descritti di seguito.

Man in the middle

Si verifica quando un utente malintenzionato si interpone nelle comunicazioni tra client e server. Questo tipo di attacco può essere molto pericoloso perché la vittima potrebbe non notare nulla.

ARP cache poisoning: l'attaccante associa il suo MAC agli indirizzi IP di altri utenti sulla rete. In questo modo, i dati della vittima vengono indirizzati all'attaccante.

Spoofing DNS: in questo modo, le richieste possono essere dirottate verso siti dannosi aventi l'aspetto del sito originale.

Falso access-point: questa tecnica prevede la creazione di un access point avente un nome simile al punto di accesso legittimo, creando un ponte tra la vittima e la rete Wi-Fi.

Man in the browser: questa tecnica prevede l'installazione di un malware sul PC della vittima che può compromettere il browser. In questo caso, un utente malintenzionato

può manipolare la pagina web mostrata all'utente richiedendo l'immissione di credenziali che verranno sottratte illegalmente.

Phishing e spear phishing

Questa è la pratica di inviare e-mail che sembrano provenire da una fonte affidabile al fine di ottenere informazioni personali o influenzare un utente ad agire. Può configurarsi in allegati di posta elettronica contenenti malware o collegamenti a siti web simili a siti ufficiali, che possono portare al download di malware o al furto di informazioni personali.

Lo spear phishing è un tipo di phishing più mirato. Gli aggressori conducono ricerche approfondite per creare messaggi personalizzati e pertinenti per un determinato utente. Per questo motivo, lo spear phishing è più difficile da identificare e quindi più difficile da prevenire.

Uno dei modi in cui viene eseguito questo tipo di attacco è lo spoofing dell'e-mail, che si verifica quando le informazioni di un mittente vengono falsificate in modo che appaiano come se provenissero da una fonte attendibile.

Per ridurre il rischio di essere vittima di phishing, è possibile utilizzare alcune tecniche:

Pensiero critico: valutare attentamente le email in arrivo analizzando le informazioni del mittente e il suo contenuto.

Verifica dei collegamenti: controllare la destinazione effettiva del collegamento.

Analisi delle intestazioni delle e-mail: le intestazioni delle e-mail definiscono il modo in cui le e-mail arrivano ad un destinatario. I parametri "Reply-to" e "Return-Path" devono puntare allo stesso dominio indicato nell'e-mail.

Sandbox: è possibile testare il contenuto di un'e-mail in un ambiente sandbox registrando l'attività di apertura degli allegati o facendo clic su un collegamento nell'e-mail.

Drive-by

Gli attacchi di drive-by download sono un metodo comune per diffondere malware. Gli attaccanti cercano di individuare siti Web non protetti e inseriscono script dannosi all'interno del codice HTML o PHP di una delle pagine. A questo punto lo script può installare malware direttamente sul computer della persona che visita il sito, oppure può reindirizzare la vittima a un sito controllato dagli attaccanti. I download drive-by possono verificarsi quando si visita un sito web, si visualizza un'e-mail o una finestra pop-up. A differenza di molti altri tipi di attacchi, il drive-by non si basa sul fatto che l'utente compia operazioni attive per innescare l'attacco. I download drive-by possono sfruttare applicazioni, sistemi operativi o browser web che contengono falle di sicurezza dovute a aggiornamenti non riusciti o mancanti.

Per proteggersi dagli attacchi drive-by, è necessario mantenere aggiornati il browser e il sistema operativo ed evitare di visitare siti web non sicuri che potrebbero contenere codice dannoso.

Cross-site scripting (XSS)

Gli attacchi XSS utilizzano risorse web di terze parti per eseguire script nel browser o nell'applicazione web della vittima. In particolare, gli aggressori iniettano payload JavaScript dannosi nei database dei siti; quando una vittima richiede una pagina da un sito web, il server trasmette la pagina insieme al payload dell'attaccante come parte del corpo HTML al browser della vittima, che esegue lo script dannoso. L'azione potrebbe consistere nell'inviare il cookie della vittima al server dell'attaccante; così facendo, l'attaccante può estrarlo e usarlo per il dirottamento della sessione. Le conseguenze più pericolose si verificano quando XSS viene utilizzato per sfruttare ulteriori vulnerabilità, in cui gli aggressori non solo possono rubare i cookie, ma anche registrare sequenze di tasti, acquisire schermate, scoprire e raccogliere informazioni di rete, accedere e ottenere il controllo remoto del PC della vittima.

XSS può essere incorporato in VBScript, ActiveX e Flash, tuttavia JavaScript è il più abusato, principalmente a causa del suo supporto diffuso sul web.

Per difendersi dagli attacchi XSS, gli sviluppatori possono convalidare i dati inseriti dall'utente nella richiesta HTTP prima della restituzione e convertire i caratteri speciali (come ?, &, /, <, >) nei rispettivi equivalenti codificati.

Malware

In generale, il malware è un software indesiderato installato nell'ambiente della vittima senza il suo reale consenso. Può connettersi ad applicazioni ampiamente utilizzate e replicarsi sulla rete. Ne esistono di diverse tipologie:

Virus macro: possono infettare applicazioni come Microsoft Word o Excel. All'apertura dell'applicazione, il virus esegue le istruzioni dannose prima di trasferire il controllo.

File infector: quando viene eseguito, infetta un file (es. file exe).

Virus di boot-record: attacca il Master Boot Record sui dischi rigidi. Quando il sistema operativo si avvia, carica in memoria il virus presente nel settore di avvio dove si diffonde ad altri dischi e computer.

Virus polimorfico: il file dannoso riesce a cambiare costantemente la sua forma sfruttando diversi cicli di crittografia e decrittografia. Tali virus sono difficili da rilevare, ma hanno un'elevata entropia a causa di notevoli mutazioni al loro codice sorgente. Il software antivirus può utilizzare questa caratteristica per rilevarli.

Virus furtivi: prendono il controllo di alcune funzioni di sistema per offuscarsi. Per fare ciò, compromettono il software di rilevamento in modo che le aree infette vengano segnalate come non infette. Quando un file viene infettato, questi virus causano un aumento delle dimensioni del file, oltre a modificare la data e l'ora dell'ultima modifica.

Trojan: codice dannoso nascosto all’interno di normali programmi. Una grande differenza tra virus e trojan è che i trojan non si replicano da soli. Oltre a lanciare attacchi ai sistemi, i trojan possono anche aprire una backdoor che può essere sfruttata dagli aggressori.

Worm: differiscono dai virus in quanto non si uniscono a un file ospite, ma sono programmi autonomi che si diffondono attraverso reti di computer. I worm vengono

generalmente diffusi tramite allegati di posta elettronica la cui apertura comporta l’attivazione del worm. Il comportamento tipico del worm prevede l'invio del proprio clone a ogni contatto di posta elettronica del computer infetto. Oltre a condurre attività dannose, i worm che si diffondono su Internet possono sovraccaricare i server di posta elettronica e possono portare ad attacchi Denial of Service ai nodi di rete.

Dropper: è un programma creato per installare malware, virus o aprire backdoor su un sistema. Il codice del malware può essere incluso nel dropper (single-stage) oppure può essere scaricato in un secondo momento (multi-stage).

Ransomware: è un tipo di malware che blocca l'accesso ai dati delle vittime, crittografandoli e minacciando di pubblicarli o cancellarli se non viene pagato un riscatto.

Adware: è un'applicazione software utilizzata dalle aziende per scopi di marketing; consiste nella comparsa di banner pubblicitari quando viene eseguito qualsiasi programma. L'adware può essere scaricato automaticamente sul sistema quando si naviga su qualsiasi sito Web e può essere visualizzato attraverso una finestra pop-up o tramite una barra (barra degli strumenti di IE) che appare automaticamente sullo schermo del computer.

Spyware: è un programma installato per raccogliere informazioni sugli utenti, i loro computer o le loro abitudini di navigazione. Tiene traccia di tutte le azioni eseguite all'insaputa dell'utente e invia i dati a un server remoto. Può anche eseguire azioni come scaricare e installare altri programmi dannosi da Internet o accedere a file e cartelle.

Code Injection

L'iniezione di codice sfrutta una vulnerabilità causata dall'elaborazione di dati non controllati; gli aggressori la utilizzano per introdurre codice che verrà eseguito in modo arbitrario. L'obiettivo di questo attacco è diffondere virus, worm, accedere a informazioni riservate, aumentare i privilegi e assumere il controllo non autorizzato del sistema. I sistemi più vulnerabili all'iniezione di codice sono SQL, LDAP, XPath, NoSQL, comandi shell e parser XML.

Per prevenire questa vulnerabilità, è necessario gestire l'input e l'output in modo sicuro, verificando la presenza di caratteri speciali che potrebbero interferire con l'esecuzione del programma. Per semplificare questo controllo, è possibile utilizzare librerie o API specializzate per ispezionare i dati di input.

SQL Injection: è una tecnica utilizzata per attaccare le applicazioni che utilizzano il linguaggio SQL per gestire i dati tramite database relazionali. Il mancato controllo dell'input dell'utente consente di inserire artificialmente stringhe di codice SQL che verranno eseguite dall'applicazione server. Grazie a questo meccanismo possono essere eseguiti anche comandi molto complessi come modifiche dei dati, download completi di database o cancellazioni totali.

SQL injection viene spesso sfruttata nell'attacco alle applicazioni web, ma viene anche utilizzata nei confronti di qualsiasi altro tipo di software che utilizza i database SQL in modo non sicuro.

Oltre a manipolare i dati, SQL injection può anche annullare le transazioni in corso, modificare gli utenti e le relative autorizzazioni.

L'Open Web Application Security Project ha affermato che negli ultimi anni questo tipo di attacco è stato considerato una delle vulnerabilità più diffuse nelle applicazioni web, rientrando nei primi dieci posti.

Tecnicamente, una caratteristica molto comune che può portare a vulnerabilità è la mancata applicazione di filtri a caratteri speciali. Questo tipo di injection si verifica quando tali caratteri, non essendo filtrati, vengono passati ad uno statement.

Ad esempio, dato:

```
statement = "SELECT * FROM users WHERE email = '"+ eMail + "';"
```

se un utente inserisce in un form dedicato all'inserimento dell'e-mail, il valore

```
' OR '6'='6
```

la query risulterà essere sempre vera, restituendo tutti i risultati.

Se questa query viene utilizzata in un'operazione di autenticazione, provocherà comportamenti imprevisti, consentendo agli utenti non autorizzati di navigare in aree protette del sito. È anche possibile eseguire più comandi contemporaneamente utilizzando commenti e separatori di istruzioni. Per motivi di sicurezza, ad oggi, alcune chiamate API al database non consentono di eseguire query con più di uno statement.

Una variante di SQL injection è chiamata blind injection e viene utilizzata in situazioni in cui l'attaccante non può vedere direttamente il risultato dell'operazione. In questo caso, in realtà, la pagina vulnerabile potrebbe non essere configurata per visualizzare i dati, ma potrebbe cambiare forma in base ai risultati logici della query. In questo caso, l'attacco sarebbe decisamente più oneroso in termini di tempo in quanto sarebbe necessario inviare una nuova istruzione per ogni bit recuperato; tuttavia esistono strumenti che automatizzano questo processo.

A livello pratico un attacco di questo tipo potrebbe essere portato a termine come segue. Supponendo di avere una pagina web che mostra le recensioni di un libro, utilizzando un url del tipo `http://book.example.com/showReview.php?ID=5` che comporta l'esecuzione della query SQL `SELECT * FROM bookreviews WHERE ID = 'Value(ID)'` la pagina web mostrerà, se presenti, i dati relativi alla recensione con ID=5. Qualora l'utente malevolo aggiunga nell'url sopra, la stringa `OR 6=6` oppure `AND 6=7` e riceva in risposta due risultati differenti, molto probabilmente la pagina web sarà vulnerabile. Questo infatti permette di capire che i due statement sono stati valutati correttamente. A questo punto il malintenzionato, sfruttando le proprietà dell'algebra booleana, potrebbe eseguire, mediante tentativi, ulteriori query, entrando in possesso di maggiori informazioni quali la versione di MySQL o altri dettagli utili al suo scopo.

Per prevenire questo tipo di attacco si può applicare un semplice controllo sui parametri passati all'istruzione. Infatti, per molte piattaforme di sviluppo, è possibile utilizzare istruzioni parametrizzate che funzionano con placeholder invece di passare direttamente l'input dell'utente. I placeholder possono memorizzare solo valori del tipo specificato e non qualsiasi stringa di testo, quindi, se passati in un formato errato, verrà generata un'eccezione.

Un'altra possibilità consiste nell'utilizzare una libreria di object-relational mapping (ORM), che gestisce la generazione automatica di istruzioni SQL parametrizzate a partire da codice orientato agli oggetti.

Esistono altre tecniche di prevenzione, come l'utilizzo di funzioni che filtrano i caratteri speciali. I manuali dei DBMS indicano quali caratteri hanno un significato speciale in SQL, il che consente di creare una lista di elementi a cui prestare attenzione.

Infine, è importante limitare i permessi dell'applicazione che accede al database per evitare che esegua operazioni potenzialmente dannose e irrecuperabili; in questo caso, anche se l'attacco ha successo, il danno sarà limitato. Alcuni esempi di operazioni rischiose includono: visualizzazione di tabelle di sistema, eliminazione di database e altri comandi simili.

Command injection: questo è un caso particolare di iniezione di codice in cui l'attaccante è limitato dalle funzionalità disponibili nel linguaggio del sistema attaccato. Si verifica quando un'applicazione passa i dati forniti dall'utente alla shell di sistema senza convalidarli. Questi comandi vengono generalmente eseguiti con le stesse autorizzazioni dell'applicazione vulnerabile.

Supponendo di avere un codice similare al seguente:

```
int main(char* argc, char** argv) {
 char cmd[CMD_MAX] = "/usr/bin/cat ";
 strcat(cmd, argv[1]);
 system(cmd);
}
```

è possibile notare che il comando system esegue un’operazione influenzata dal parametro argv[1]. In questo caso possiamo assumere che in condizioni normali il contenuto di tale parametro sia legittimo ed utile allo scopo del programma; tuttavia un utente malintenzionato potrebbe, senza alcuna difficoltà, procedere alla modifica del parametro con un contenuto malevolo. Il programma procederà quindi ad eseguire e mostrare all’attaccante tutte le operazioni da lui impartite, ad esempio potrebbe accedere a file di sistema, file contenenti password, nonché procedere alla modifica o cancellazione delle informazioni fino a prendere il controllo totale dell’host. Se il programma eseguiva con privilegi di amministrazione, anche eventuali comandi

malevoli saranno eseguiti con gli stessi privilegi. Se un malintenzionato inserisse come parametro argv[1] il valore `";rm -rf /"` la chiamata system procederà alla cancellazione ricorsiva della partizione root. Nel caso in esempio le tipologie di comando iniettabili sono esclusivamente quelle ammesse nel sistema operativo in uso, la scrittura di comandi non previsti genererà un errore.

Durante la fase di programmazione è necessario prestare attenzione all'utilizzo di comandi in grado di comunicare con altre applicazioni, in quanto questi comandi potrebbero non utilizzare il sistema di convalida degli input. Se questa funzionalità non è garantita, deve essere implementata dal programmatore. Ad oggi, la maggior parte delle applicazioni fornisce API per comunicare con altri programmi in modo sicuro ed efficiente, riducendo notevolmente tali vulnerabilità.

2.1 Denial of service (DoS) e distributed denial of service (DDoS)

Il Denial of Service è un tipo di attacco progettato per sovraccaricare le risorse di sistema e renderle non disponibili agli utenti reali. Il DDoS si differenzia dal primo per come viene implementato, il secondo infatti è eseguito da un gran numero di macchine, solitamente geograficamente distribuite.

È anche possibile che gli attacchi DDoS vengano eseguiti da utenti inconsapevoli perché sono stati precedentemente infettati da malware controllato da un aggressore. A differenza di altri tipi di attacchi finalizzati al guadagno finanziario diretto, in molti casi DoS e DDoS mirano esclusivamente a creare un disservizio. Tuttavia, in alcuni casi vi sono vantaggi immediati, come la disabilitazione di un servizio che viene sfruttato per eseguire un'altra azione dannosa (session hijacking).

Esistono diversi modi per eseguire attacchi DoS e DDoS:

TCP SYN Flood: un utente malintenzionato sfrutta l'uso dello spazio del buffer durante l'handshake di inizializzazione della sessione TCP. Il dispositivo dell'attaccante inonda la coda in-process del sistema di destinazione (solitamente limitata) con richieste di connessione senza inviare alcun riscontro a tali richieste. Questo fa sì che il sistema

di destinazione vada in timeout mentre aspetta la risposta dal dispositivo dell’attaccante. Una volta che la coda di connessione si riempie, il sistema diventa non disponibile. Le possibili soluzioni includono la protezione del server dai pacchetti SYN in entrata utilizzando un firewall, l'aumento delle dimensioni della coda e la riduzione del timeout sulle connessioni aperte.

Smurf: questo attacco comporta la saturazione della rete di destinazione mediante IP spoofing e ICMP. A partire dall'indirizzo vittima di spoofing, vengono inviate richieste ICMP echo all'IP broadcast, saturando la rete. Questo processo è ripetibile e può generare automaticamente una grave congestione. Una possibile soluzione è disabilitare il broadcast diretto o configurare gli endpoint per impedire loro di rispondere ai pacchetti ICMP provenienti dagli indirizzi di broadcast.

Botnet: consiste in una rete di computer controllata da un botmaster, costituita da dispositivi infettati da malware. Ogni computer nella botnet si connette a una risorsa del centro di comando tramite un dominio web o un canale IRC per ricevere istruzioni. Più recentemente, le botnet hanno adottato un modello peer-to-peer per eliminare il singolo punto di errore che esiste sui server centralizzati, in modo che ogni bot possa fungere da client e server per altri nodi e diffondere i dati.

3 L’importanza dei log nei software applicativi

I file di log registrano gli eventi che si verificano durante l'esecuzione del software. Nel caso più semplice, i log vengono annotati in un unico file. Molti sistemi operativi, framework software e programmi includono un sistema di logging che può utilizzare standard di registrazione. Uno di questi standard è syslog, definito in Internet Engineering Task Force (IETF) RFC 5424, che consente a sottosistemi standardizzati dedicati di generare, filtrare, registrare e analizzare i messaggi di log. Ciò esonera gli sviluppatori di software dal dover progettare e configurare sistemi di registrazione proprietari.

I log possono essere di vario tipo, inclusi eventi, transazioni, messaggi, server ed errori.

I log degli eventi registrano ciò che si verifica durante l'esecuzione del programma per fornire un audit trail che può essere utilizzato per monitorare l'attività e diagnosticare i problemi. Sono particolarmente importanti per le applicazioni con poca interazione con l'utente.

La maggior parte dei DBMS utilizza i log delle transazioni per registrare le modifiche apportate ai dati archiviati. Ciò consente al database di riprendere le sue operazioni e mantenere la coerenza dei dati dopo un arresto anomalo o altri errori. A tale scopo, il DMBS dispone di log degli eventi e delle transazioni.

I log dei messaggi sono generalmente rappresentati come file di testo, ma alcuni client possono salvarli come file HTML o altri formati personalizzati per una gestione più semplice. Questo tipo di log viene utilizzato per Internet Relay Chat (IRC), programmi di messaggistica istantanea e chat di videogiochi. Per il software IRC, i log includono messaggi di sistema e record relativi alle modifiche dell'utente, come la connessione o disconnessione dalle chat.

I log dei server vengono creati e gestiti dal server per le attività che esegue. Un uso tipico dei log del server web consiste nel mantenere una cronologia delle pagine richieste. Il formato standard per tali log è chiamato Common Log Format, creato dal W3C, ma esistono anche altri formati proprietari. Questo standard specifico memorizza

informazioni come l'indirizzo IP del cliente, la data e l'ora della richiesta, la pagina richiesta, il codice HTTP, lo user-agent e il referrer. Tali log non contengono informazioni specifiche dell'utente e sono accessibili solo agli amministratori del sito o ad altro personale autorizzato.

I log degli errori registrano gli errori critici riscontrati da un'applicazione, un sistema operativo o un server durante l'esecuzione. Questi sono utili per la risoluzione dei problemi e la gestione dei sistemi server e delle reti; possono contenere tutte le informazioni sull'errore o solo alcuni codici di errore specifici. In molti casi, l'accesso a questi registri richiede autorizzazioni speciali perché possono essere utilizzati come misura di sicurezza per impedire l'accesso a risorse non autorizzate. Nel caso di server e reti di uffici questi log tengono traccia delle problematiche riscontrate dagli utenti e aiutano l’analisi per la loro risoluzione.

3.1 Gli scopi dei log in un DBMS

Il DBMS gestisce i log utilizzando file sequenziali che registrano le operazioni di modifica eseguite dalle transazioni. Questo è fondamentale per garantire la persistenza a fronte di Transaction Failure e System Failure. I record vengono scritti dopo ciascuna delle seguenti operazioni sui file di log mediante l’uso di tabelle.

Begin	quando una transazione inizia.
Update	aggiornamento di una pagina. Avviene quando una transazione rende “sporca” una pagina.
Commit	completamento corretto di una transazione.
Abort	mancato completamento (Abort) di una transazione
End	terminazione di una transazione (successiva al commit/abort) cioè quando i dati vengono effettivamente resi permanenti sul

	disco
Compensation	registra l'annullamento degli aggiornamenti di una transazione, ad esempio quando una transazione abortisce (per rollback o abortita dal sistema) bisogna disfarsi delle modifiche effettuate dalla transazione stessa, quindi si effettuano delle modifiche a ritroso per riuscire a recuperare i valori iniziali.

Un file di log è costituito da record che hanno una struttura specifica in base alle operazioni eseguite: di seguito un esempio di record di update e record di compensazione. Il formato del record di update della transazione T che modifica la pagina del database P è il seguente:

```
(LSN, prevLSN, T, type, PID, before(P), after(P))
```

LSN	Log Sequence Number è un numero progressivo del record (identifica il record)
prevLSN	identifica il LSN del precedente record del LOG relativo alla transazione T, in modo da avere i record di una stessa transazione collegati a lista
T	identificatore della transazione
type	è il tipo del record, update in questo caso
PID	identificatore della pagina modificata
before(P)	"before image" di P, ovvero il contenuto della pagina P prima della modifica (utile per annullare le modifiche di una transazione abortita)
after(P)	"after image" di P, ovvero il contenuto della pagina P dopo la modifica (utile per ripristinare le modifiche di una transazione terminata con successo ma i cui dati non sono stati resi

persistenti)

I record di compensazione vengono utilizzati quando il risultato di un'operazione di modifica viene annullato, ad esempio nel caso di abort di una transazione. Il formato del record di compensazione per la transazione T è il seguente:

`(LSN, prevLSN, T, type, undoNextLSN, PID, before(P))`

LSN	Log Sequence Number
prevLSN	identifica LSN del precedente record del LOG relativo alla transazione T
T	identificatore della transazione
type	è il tipo del record, compensation in questo caso
undoNextLSN	rappresenta il prossimo record da annullare: se stiamo annullando il record U corrisponde al prevLSN di U (poiché stiamo operando a ritroso)
PID	identificatore della pagina modificata
before(P)	“before image” di P, ovvero il contenuto della pagina P prima della modifica.

Di seguito viene mostrato un esempio di log più esplicativo.

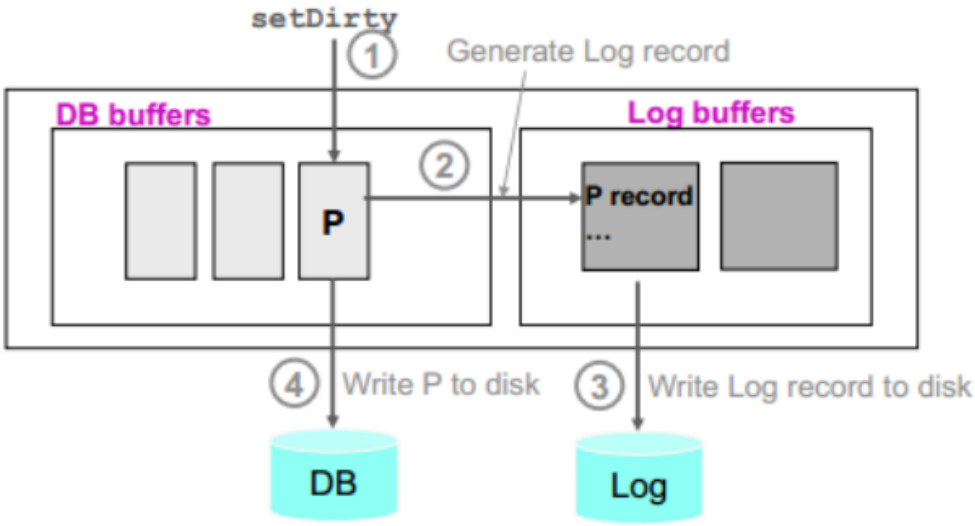
LSN	prevLSN	T	type	PID	before(P)	after(P)
...						
235	-	T1	BEGIN			
236	-	T2	BEGIN			
237	235	T1	UPDATE	P15	(abc, 10)	(abc, 20)
238	236	T2	UPDATE	P18	(def, 13)	(ghf, 13)
239	237	T1	COMMIT			
240	239	T1	END			
241	238	T2	UPDATE	P19	(def, 15)	(ghf, 15)
242	-	T3	BEGIN			
243	241	T2	UPDATE	P19	(ghf, 15)	(ghf, 17)
244	242	T3	UPDATE	P15	(abc, 20)	(abc, 30)
245	243	T2	ABORT			
246	244	T3	COMMIT			
247	243	T2	END			
...						

Come si può notare i record 235, 237, 239 e 240 forniscono una lista (attraverso i prevLSN) di tutte le modifiche effettuate dalla transazione T1.

Protocollo WAL: per poter utilizzare il log per ripristinare lo stato del database in caso di guasto, è importante applicare il cosiddetto protocollo WAL (Write-Ahead Logging): prima che una pagina modificata P venga scritta su disco, i record di log corrispondenti devono essere stati scritti nel log. Intuitivamente, se non si segue il protocollo WAL, potrebbero accadere le seguenti anomalie:

1. una transazione T modifica il database aggiornando una pagina P
2. prima di scrivere il log record sul log relativo alla modifica di P, avviene un System Failure.

In questa situazione, ovviamente, non è possibile ripristinare in alcun modo il database allo stato originale, perché non ci sono informazioni sufficienti per poter eseguire un "rollback" allo stato originale. La responsabilità di garantire la conformità con il protocollo WAL spetta al Buffer Manager che gestisce, oltre ai buffer di database, i buffer di log (differenti dai buffer di database). Il diagramma seguente mostra l'ordine in cui avvengono le varie operazioni relative alla modifica di una pagina P.



L'esecuzione dunque si suddivide in:

1. una transazione avvisa il Buffer Manager che la pagina P è sporca (attraverso il setDirty) poiché è stata modificata;
2. il Buffer Manager accede ai buffer del log e scrive il record di log relativo alla modifica sul log stesso;
3. il log precedentemente caricato in memoria può essere scritto su memoria stabile (anche in un momento successivo);
4. infine, a seconda delle politiche attuate dal Buffer Manager, la modifica della pagina P viene resa persistente e scritta su disco.

Come accennato in precedenza, il log deve essere scritto su una memoria stabile perché deve sopravvivere sia ai guasti di sistema, sia a quelli dei dispositivi. La memoria stabile si ottiene conservando copie delle informazioni su dispositivi permanenti (possibilmente in luoghi diversi), utilizzando tecniche RAID, mirroring e bit di parità. Inoltre, si fa notare che il disco dedicato ai log non è lo stesso di quello dedicato al database: oltre ad essere una memoria stabile, essendo il log un file sequenziale, la latenza è molto bassa poiché la testina non ha bisogno di essere spostata; questa rimane fissa nel punto in cui è posizionata ed è sempre pronta a scrivere nel blocco successivo all'ultimo. Per questo motivo è meglio avere un disco affidabile piuttosto che uno molto performante. Il log permette al Recovery Manager di annullare le operazioni sulle transazioni interrotte e

incomplete e di rieseguire le operazioni sulle transazioni che sono state completate (committed) con successo ma le cui modifiche non sono ancora state rese consistenti.

Dato che bastano i log record per recuperare le informazioni necessarie, si precisa che una transazione può essere definita commited solo quando i suoi log record sono stati scritti su memoria stabile.

Il log può risolvere il Transaction Failure e il System Failure nei seguenti modi:

Transaction Failure: adottando la politica steal (cioè la pagina P viene scritta quando “conviene”), se una transazione T abortisce, è possibile che alcune delle pagine da essa modificate siano state già scritte su disco. Bisogna dunque disfarsi delle modifiche della transazione abortita: per annullare queste modifiche si scandisce il log a ritroso usando i prevLSN e si ripristinano nel database le before image delle pagine modificate da T. Nell’esempio seguente viene mostrato l’annullamento delle modifiche effettuate dalla transazione T2 che abortisce.

LSN	prevLSN	T	type	PID	before(P)	after(P)
...						
236	-	T2	BEGIN			
237	235	T1	UPDATE	P15	(abc, 10)	(abc, 20)
238	236	T2	UPDATE	P18	(def, 13)	(ghf, 13)
239	237	T1	COMMIT			
240	238	T2	UPDATE	P19	(def, 15)	(ghf, 15)
241	-	T3	BEGIN			
242	240	T2	UPDATE	P19	(ghf, 15)	(ghf, 17)
243	241	T3	UPDATE	P15	(abc, 20)	(abc, 30)
244	242	T2	ABORT			

System Failure: in caso di un System Failure, applicando lo stesso algoritmo del Transaction Failure, vengono rieseguite tutte le transazioni per le quali non si trova il record COMMIT sul LOG, questo perché erano transazioni attive di cui non si conoscono gli esiti.

Adottando la politica no-force (all’atto del commit si scrive solo sul log e non è

garantita la scrittura immediata su disco delle modifiche), una transazione T terminata correttamente non ha certezza di aver reso permanenti le modifiche sul disco, pertanto occorre rieseguire gli stessi passi che T aveva fatto durante la sua esecuzione, riscrivendo le after image che si trovano sul log. Nell'esempio seguente viene mostrato il rifacimento della transazione T1.

LSN	prevLSN	T	type	PID	before(P)	after(P)
...						
235	-	T1	BEGIN			
236	-	T2	BEGIN			
237	235	T1	UPDATE	P15	(abc, 10)	(abc, 20)
238	236	T2	UPDATE	P18	(def, 13)	(ghf, 13)
239	237	T1	COMMIT			
...						

In entrambi i casi precedenti, non si era certi se la pagina fosse stata modificata o meno. Infatti, nel caso di una transazione abortita, non si ha la certezza che alcune o tutte le pagine siano state effettivamente scritte su disco, così come nel caso di una transazione committed non si è certi se alcune pagine debbano ancora essere rese permanenti. Di conseguenza si verifica una riscrittura non necessaria che porta ad inefficienza. Per evitare di riscrivere l'after image delle pagine modificate da tutte le transazioni committed, il Buffer Manager adotta le seguenti precauzioni: quando la pagina P viene modificata dalla transazione T, genera un record di log al quale assegna il proprio LSN; quindi scrive l'LSN insieme al PID nel page header di P. Si veda la figura.

Pagina P15 su disco

Page Header	PID	LSN
	P15	293

LSN	prevLSN	T	type	PID	before(P)	after(P)
...						
237	...	T1	UPDATE	P15	(abc, 10)	(abc, 20)
238	...	T2	UPDATE	P15	(abc, 20)	(ghf, 13)
...						
327	...	T3		P15	(ghf, 13)	(ghf, 18)
...						

Indichiamo con LSN(P) l'identificativo salvato sul page header della pagina P e con k l'LSN di un log record relativo alla pagina P: quando la transazione T viene rieseguita, se LSN(P) ≥ k allora non è necessario riscrivere la pagina P (cioè non

è necessario aggiornare con after image del log record) poiché si è certi che le modifiche di questo record erano già state rese permanenti su disco. Quindi è vero che si leggono tutte le pagine modificate dalla transazione T, ma vengono riscritte solo quelle necessarie. Un discorso analogo, ma applicato all'inverso, può essere fatto in caso di transazione abortita.

Checkpoint: il processo di restart è responsabile del ripristino del database a uno stato consistente in caso di System Failure. Per ridurre i tempi di riavvio, è possibile eseguire "checkpoint" periodici, che forzano la scrittura su disco di tutte le pagine modificate.

LSN	prevLSN	T	type	PID	before(P)	after(P)
237	...	T3	UPDATE	P15	...	...
238	...	T2	UPDATE	P18	...	...
239	...	T1	UPDATE	P17	...	...
240	...	T1	COMMIT			
241	...	T2	COMMIT			
242			CKP			
243	...	T3	UPDATE	P19	...	...

L'esecuzione dei checkpoint viene registrata nel log grazie al record CKP che comprende la tabella delle transazioni attive e quella delle pagine "sporche". I record di checkpoint contengono informazioni diverse rispetto ai record visti in precedenza, il che è inevitabile perché, per essere in grado di capire quali informazioni sono state salvate e quali no, i record di checkpoint necessitano di informazioni aggiuntive, come le transazioni attive e le pagine sporche (in caso di System Failure si conoscono esattamente quali pagine e quali transazioni controllare). In questo modo, se la transazione T ha eseguito il commit prima del checkpoint, si è sicuri che T non debba essere rifatta, poiché si è certi che le pagine della transazione T siano state scritte su disco.

4 Sessioni e Cookie nelle WebApplication

Le applicazioni e i siti web hanno bisogno di memorizzare dei dati associati ad ogni client che si collega. Un modo per memorizzare questi dati direttamente all'interno del client web sono gli HTTP Cookie che hanno molti utilizzi, tra cui salvare le abitudini dell'utente all'interno dei siti web, utilizzarli per la pubblicità mirata o per salvare i dati di una sessione per evitare la necessità di una nuova autenticazione in una successiva visita. Della creazione dei Cookies si occupa il sito web visitato e ogni visita verso quel sito contiene i precedenti Cookies creati. All'inizio del loro utilizzo, i Cookies, salvavano molti dati riguardanti l'utente che, ad ogni richiesta, dovevano essere trasferiti dal client al server e viceversa causando un overhead di rete. Per risolvere questo problema si è deciso di memorizzare all'interno del Cookie solamente un ID di sessione univoco che consente di salvare solamente nel server i dati associati.

Funzionamento dei Cookies

I Cookies sono contenuti nell'header HTTP sia della richiesta sia della risposta. Nel caso della richiesta i Cookies sono preceduti dalla parola chiave "Cookie:", invece nella risposta dal server sono preceduti da "Set-Cookie:". Nel momento in cui un browser riceve un header contenente Cookies si occupa di memorizzarli sottoforma di file all'interno di una directory dedicata. Ogni volta che il browser carica un determinato sito web, rimanderà i Cookies a lui associati senza modificarli e nella risposta, il server, potrà eventualmente crearne di nuovi o modificare quelli esistenti.

Essendo HTTP un protocollo stateless e visto che le pagine web devono avere un buon grado di dinamicità in base alle interazioni dell'utente, grazie al meccanismo dei Cookies, è possibile aggiungere lo stato al protocollo. Infatti senza i Cookies non vi sarebbe differenza in una pagina caricata prima o dopo un login. Dato che i Cookies permangono nel sistema per lunghi periodi, i siti possono assegnare un indice all'utente e tenere traccia della sua navigazione all'interno del sito, solitamente allo scopo di creare statistiche o fornire consigli personalizzati.

Struttura dei Cookie

I Cookies sono composti da:

- una stringa di testo che ne rappresenta il valore;
- il dominio di competenza:
- il percorso al quale rinviarli all'interno del dominio;
- una data di scadenza dopo la quale non sono più validi e il browser può eliminarli;
- un tag HttpOnly che lo rende leggibile solo tramite le richieste HTTP;
- un tag Secure che indica se deve essere trasmesso criptato con HTTPS;
- un tag SameSite che consente l'invio del Cookie solo per richieste provenienti dalla stessa fonte per neutralizzare gli attacchi come CSRF.

I Cookies possono essere di sessione o persistenti. Nel primo caso non sono dotati di data di scadenza, per questo non vengono memorizzati in modo persistente nel dispositivo dell'utente e si cancellano alla chiusura del browser. Nel secondo caso, invece, sono dotati di data di scadenza che li rende validi fino a quel momento.

I Cookies si classificano di prima parte nel caso in cui il dominio sia lo stesso del sito web visitato o di terza parte nel caso in cui le pagine web visitate contengano contenuti provenienti da altri siti come gli annunci pubblicitari. Questa categoria di Cookies è spesso utilizzata dagli inserzionisti allo scopo di servire annunci personalizzati e rilevanti per ciascun utente.

Infine è possibile distinguere diverse tipologie di Cookie a seconda della loro finalità:

- Cookie tecnici: servono per la navigazione e per facilitare l'accesso e la fruizione del sito da parte dell'utente. Sono essenziali per ottenere un comportamento user-friendly della pagina web e garantire l'erogazione corretta del servizio;
- Cookie statistici: vengono utilizzati a fini di ottimizzazione del sito, direttamente dal titolare del sito stesso, che potrà raccogliere informazioni in forma aggregata sul numero degli utenti e su come questi visitano il sito;
- Cookie per la memorizzazione delle preferenze (o funzionali): sono utili a favorire l'utilizzo efficace del sito da parte dell'utente e favorire così l'esperienza personalizzata di navigazione. Vengono utilizzati, ad esempio, per tenere

traccia della lingua scelta;

- Cookie di marketing e profilazione (pubblicitari): questi hanno lo scopo di fornire spazi pubblicitari; possono essere applicati, dal titolare del sito oppure da terze parti e vengono utilizzati per studiare il “profilo” di navigazione dell'utente, in modo da poter proporre messaggi pubblicitari in linea al suo comportamento e interessi nella rete;
- Cookie di social network: si tratta dei Cookies che consentono di condividere anche con altri utenti i contenuti del sito che si sta visitando. Queste funzioni consentono ai social network di identificare i propri utenti e raccogliere informazioni anche mentre navigano su altri siti.

Privacy e Cookie

A partire dal 25 maggio 2018 anche in Italia è entrato in vigore il regolamento generale sulla protezione dei dati (GDPR) che stabilisce che anche i Cookies dei siti web sono dati personali e come tali devono essere soggetti a regole sul trattamento. Nello specifico devono essere gestiti come segue:

- i dati personali non possono essere tracciati o utilizzati prima che l’utente ne abbia dato un consenso esplicito;
- su tutte le pagine web del sito devono essere specificati i tracciamenti applicati;
- gli utenti devono essere informati in linguaggio semplice e comprensibile su chi riceve i loro dati, come li utilizza e della durata del trattamento;
- le autorizzazioni dell’utente devono essere registrate in modo tale che possano essere dimostrate all’autorità in caso di contenzioso;
- la revoca del consenso deve essere sempre concessa e facile da attuare anche in momenti successivi.

La Cookie Policy dei siti web deve specificare le tipologie di Cookies presenti nel sito web ad eccezione del caso in cui il sito web faccia uso esclusivamente di Cookies tecnici o funzionali.

Manipolazione dei Cookie

I Cookies essendo salvati nel browser dell'utente sono facilmente modificabili. Una tecnica di manipolazione è la Cookie Poisoning che consiste nel modificare i contenuti di un Cookie per eludere i meccanismi di sicurezza, ottenere informazioni private o prendere possesso dell'identità di un altro utente. La maggior parte dei siti web utilizzano i Cookies per identificare le sessioni dei clienti che hanno effettuato l'accesso al sito, in questo modo un attaccante si può impossessare del valore dell'identificatore della sessione e far sembrare al sito di essere un'altra persona, potendo accedere al suo account.

Per impossessarsi dei Cookie ci possono essere vari metodi tra cui il più semplice è avere l'accesso diretto al browser; un altro metodo, leggermente più avanzato riguarda il caso in cui la vittima si sia connessa tramite il protocollo HTTP non sicuro dove il contenuto dei pacchetti, compresi i Cookies, non sono criptati e un attaccante connesso alla stessa rete può intercettare questi dati senza essere rintracciato. Questo problema può essere risolto utilizzando una comunicazione cifrata come il protocollo HTTPS o specificando il flag Secure nelle impostazioni del Cookie per essere inviato solo su canali crittografati.

Un altro metodo è il DNS Poisoning che, nel momento in cui un utente cerca di collegarsi ad un dominio, reindirizza lo stesso utente all'IP dell'attaccante, trasmettendogli direttamente i Cookies. Anche questo caso si può risolvere utilizzando la connessione sicura per connettersi al DNS.

Sessioni

Come già detto, salvare tutti i dati di un certo account nel browser può essere dispendioso, quindi si è deciso di tenere solo un identificativo di sessione. La sessione viene mantenuta dal momento in cui un utente accede ad un sito fino al momento in cui effettua il logout o chiude il browser.

Essendo le sessioni mantenute sul server web, queste vanno ad occupare le sue risorse; per questo gli amministratori del server o del sito possono decidere di limitare il numero massimo di sessioni attive nello stesso momento onde evitare di occupare tutte le risorse. Per evitare che un utente occupi una sessione per un tempo

indeterminato senza utilizzarla effettivamente, si è deciso di impostare un timeout dopo il quale la sessione viene eliminata liberando una risorsa.

I vantaggi delle sessioni riguardano l'aggiunta di uno stato al protocollo HTTP come i Cookies; in questo modo, salvando temporaneamente nel browser solo un dato, viene ridotta la possibilità di modificare altri parametri associati all'account.

Per mantenere lo stato di sessione nel protocollo HTTP vi sono diverse tecniche: alcune utilizzano i Cookies; altre memorizzano le sessioni all'interno di parametri dell'URL in caso di richieste GET oppure in argomenti del corpo su richieste POST. Il meccanismo di scambio dell'ID di sessione dovrebbe comunque essere il più sicuro possibile, quindi includerlo nell'URL potrebbe farlo rivelare nei registri web o nella cronologia.

Ad occuparsi della generazione e della gestione degli ID di sessione è il server o framework web che crea una stringa casuale e univoca e la associa a un determinato utente.

Criticità delle sessioni

Come i Cookies anche le sessioni soffrono di alcune criticità.

Per la generazione delle sessioni ci sono delle politiche gestite dall'amministratore del sito. In un caso normale, quando un utente accede per la prima volta al sito dopo l'apertura del browser, invia una richiesta che non contiene alcuna sessione, quindi il server genera e associa una sessione.

Il server, nella gestione delle sessioni, deve anche verificare che gli ID che riceve dagli utenti siano realmente presenti e ancora validi. Qualora esso riceva un ID non valido, poiché scaduto o manomesso, di norma procede alla generazione di un nuovo identificativo, restituendolo all'utente.

Un utente malintenzionato, sapendo che il numero di sessioni è limitato, potrebbe adottare comportamenti anomali con lo scopo di esaurire tutte le sessioni applicative e rendere il servizio indisponibile ai reali utilizzatori. Ad esempio, un attaccante potrebbe pensare di inondare il server web trasmettendo richieste contenenti

SessionID non validi, alle quali il server risponderà con altrettante sessioni generate occupando di fatto gli slot disponibili. Nel momento in cui tutte le sessioni di un determinato servizio saranno allocate, questi smetterà di rispondere causando un disservizio.

In maniera analoga un malintenzionato, invece di trasmettere richieste contenenti un SessionID non valido, potrebbe inviare richieste non dotate di identificativo; allo stesso modo il server procederà alla generazione delle sessioni fino all'esaurimento di tutte le risorse disponibili.

4.1 Funzionamento nei linguaggi web

In questo paragrafo viene approfondito come vengono gestiti Cookie e Sessioni all'interno di alcuni server web.

Uno dei webserver più diffusi è Apache Tomcat, un progetto Open Source sviluppato dalla Apache Software Foundation che implementa le specifiche Java Server Page (JSP) e Servlet, fornendo una piattaforma per l'esecuzione di applicazioni web sviluppate in java.

Supponendo di avere una web application basata su Servlet e JSP, alla prima richiesta di caricamento, Tomcat invoca il metodo `init()` che si occupa di inizializzare i parametri necessari, come la connessione a un database. Durante la sua esecuzione, ad ogni richiesta HTTP viene chiamato il metodo `service()` che invoca a sua volta `doGet()` o `doPost()` a seconda del tipo di richiesta ricevuta, infine quando la servlet deve essere disattivata viene chiamato `destroy()`.

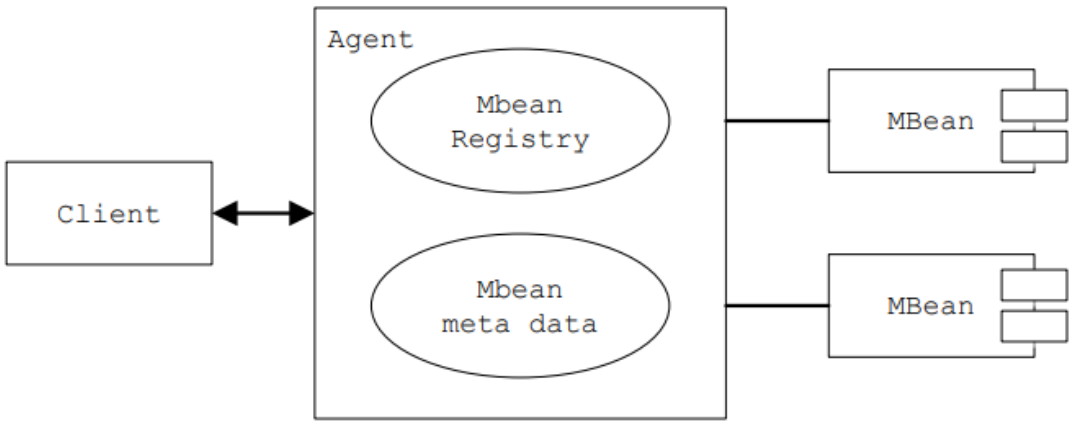
Sia la funzione `doGet()` sia `doPost()` ricevono come parametri la richiesta e la risposta. Nel caso della richiesta si usa la funzione `getParameter()` per accedere al parametro di interesse e la funzione `getCookies()` per ottenere un array contenente tutti i Cookies che il client ha inviato. Se si vuole ricevere la sessione relativa a una richiesta si può chiamare il metodo `getSession()` passandogli un valore booleano che indica se creare o meno una nuova sessione qualora non

esista.

Una volta ottenuta la sessione vi sono dei metodi specifici da cui si può conoscere il suo ID, se è stata appena creata, da quanto tempo è attiva e il suo ultimo utilizzo; eventualmente, può anche essere invalidata.

Nel caso della risposta alcuni dei dati che possono essere gestiti e che verranno restituiti al client saranno il codice di stato, l'header della risposta HTTP e il body. Per inserire i dati all'interno del body bisogna ottenere un OutputStream in cui scrivere il contenuto, ad esempio una pagina HTML. Eventualmente, si possono impostare dei Cookies con la funzione `addCookie()` o impostare un reindirizzamento tramite `sendRedirect()` che si occupa anche di cambiare il codice di stato per indicare il reindirizzamento.

JMX (Java Management Extension) è una tecnologia per il monitoraggio, controllo e gestione dei componenti Java. Questa tecnologia non produce logica di business ma viene utilizzata per controllare a runtime come si comporta tale logica. JMX permette di eseguire controllo on-line e conseguenti azioni di gestione come: misurazioni delle performance, fault tolerance, identificazione di colli di bottiglia e determinazione dei punti critici. Il componente principale di JMX è chiamato Mbean e rappresenta un oggetto che deve essere monitorato e gestito.



L'architettura di JMX è basata su tre livelli: instrumentation (insieme degli Mbean), agente e servizi distribuiti. Il livello agente è posizionato al di sopra di instrumentation ed è costituito dal registro degli Mbean e da alcuni server

aggiuntivi; il livello dei servizi distribuiti invece è formato da due supporti: i connettori e gli adattatori di supporto, che permettono di collegare l'Mbean server ai vari clienti.

Grazie a questa architettura è possibile monitorare costantemente lo stato di una Web Application e notificare eventuali anomalie. Nello specifico si può utilizzare JMX per tenere monitorata la quantità di sessioni applicative allocate in ogni istante e di conseguenza attuare politiche di gestione del traffico.

Un altro linguaggio web molto diffuso è PHP che, come gli altri, dà la possibilità di gestire Cookie e sessioni. In particolare, per gestire le sessioni, bisogna inizializzarle tramite la funzione `session_start()` a cui possono essere eventualmente passati dei parametri che modificano il comportamento di default. Una volta inizializzata la sessione, le informazioni vengono salvate all'interno dell'array `$_SESSION`. Se si decide di rimuovere un valore dall'array questo può essere passato alla funzione `unset()` (es. `unset($_SESSION['id'])`); qualora si volessero rimuovere tutti i dati della sessione si può invocare `session_unset()`. Una volta che la sessione non è più necessaria può essere eliminata tramite `session_destroy()`. Quando viene richiesta un'altra pagina, la sessione non viene passata automaticamente, ma è necessario chiamare nuovamente `session_start()` che andrà a recuperare i dati.

5 Intelligenza Artificiale per la Cybersecurity

Non esiste un metodo migliore di un altro per proteggersi dagli attacchi, ma, esistendone di vari tipi, ogni metodo è più o meno specifico per una determinata situazione.

Il metodo più semplice è mantenere il software aggiornato partendo dal sistema operativo fino ad arrivare ai programmi applicativi e i loro plugin. Mantenere i software aggiornati però non è sufficiente per ritenersi sicuri; per questo viene in aiuto l'intelligenza artificiale.

Ci sono vari tipi di intelligenza artificiale: quella ristretta, progettata e costruita per uno scopo specifico e quella generale, per apprendere e risolvere qualsiasi problema presentato. Per allenare l'IA ci sono due metodi: Machine Learning e il Deep Learning; in entrambi i casi esistono degli algoritmi che consentono di apprendere basandosi su dati passati e applicare su altri dati ciò che hanno appreso. L'apprendimento è più preciso nel caso in cui sia possibile delimitare il contesto, avere delle regole ben definite e una grande quantità di dati di buona qualità.

I sistemi basati sull'IA presentano caratteristiche specifiche che possono essere attaccate in modi non tradizionali, in particolare il set di dati di addestramento può essere compromesso in modo tale che l'apprendimento del sistema non sia quello previsto; in alternativa quello che verrà rilevato può essere manomesso in modo che il sistema non lo riconosca. È quindi importante fornire una protezione aggiuntiva ai sistemi di IA per garantire che seguano un ciclo di vita e di sviluppo sicuro, dall'ideazione all'implementazione e alla sorveglianza post commercializzazione.

Il Machine Learning utilizza algoritmi e classificatori statistici tradizionali, ad esempio:

- algoritmi di regressione;
- algoritmi bayesiani;
- classificatori;
- algoritmi basati su istanze (k-Nearest Neighbour, Support Vector Machine);
- Decision Tree;

- Clustering;
- reti neurali (Perceptron, Back propagation, Hopfield networks)

Mentre gli algoritmi di Machine Learning tradizionali sono lineari, quelli di Deep Learning sono impilati in una gerarchia di crescente complessità e astrazione.

Per il Machine Learning esistono tre paradigmi principali: reinforcement learning, supervised learning, unsupervised learning.

Reinforcement learning

Questo paradigma si occupa di risolvere problemi riferiti a decisioni sequenziali, in cui l'azione da compiere dipende dallo stato attuale del sistema e ne determina quello futuro. La qualità di un'azione è data da un valore numerico di ricompensa che ha lo scopo di incoraggiare comportamenti corretti. Il progettista stabilisce la politica di ricompensa, ma non fornisce al modello suggerimenti sulla risoluzione del problema. In questo modo è il modello a dover capire come eseguire le azioni per massimizzare la ricompensa, iniziando con prove casuali e finendo con tattiche sofisticate e abilità anche superiori alle scelte umane. Se questo viene eseguito su una potente infrastruttura computazionale si possono raccogliere esperienze da un grandissimo numero di prove parallele. Questo è il metodo più efficace per rendere la macchina autonoma e creativa.

Apprendimento supervisionato

È una tecnica che mira a istruire una rete in modo da effettuare previsioni sui valori di uscita rispetto a un input su una base di dati ideale costituita da coppie input - output che vengono inizialmente fornite.

Questi algoritmi partono dal presupposto che se si fornisce al sistema un numero adeguato di esempi, questo accumulerà un'esperienza sufficiente in modo da permettergli di creare una funzione per approssimare il comportamento desiderato in base agli esempi forniti.

Apprendimento non supervisionato

È una tecnica di apprendimento automatico che consiste nel fornire al sistema una serie di input che verranno riclassificati e organizzati sulla base di caratteristiche comuni per cercare di effettuare ragionamenti e previsioni sugli input successivi. A differenza dell'apprendimento supervisionato vengono forniti solo esempi non annotati, in quanto le classi non sono note a priori, ma devono essere apprese automaticamente. Durante la fase di apprendimento, una rete non supervisionata cerca di imitare i dati che gli vengono forniti e usa l'errore nell'output per correggere i pesi e distorsioni. Uno tra i metodi usati per l'apprendimento non supervisionato è l'analisi dei cluster usata per raggruppare o segmentare i set di dati con attributi condivisi per estrapolare relazioni algoritmiche.

L'analisi dei cluster serve per raggruppare i dati che non sono etichettati, classificati o categorizzati. Quando riceve un dato reagisce in base alla presenza o assenza dei punti in comune nel nuovo dato. Questo approccio aiuta a rilevare dati che non rientrano nelle categorie trovate.

Deep Learning

Con il Deep Learning vengono simulati i processi di apprendimento del cervello biologico attraverso sistemi artificiali per insegnare alle macchine non solo ad apprendere autonomamente ma a farlo in modo più profondo, dove profondo significa su più livelli (vale a dire su un numero di layer nascosti nella rete neurale chiamati hidden layer). Le reti di base contengono due o tre layer, mentre quelle profonde possono arrivare anche a centocinquanta. Partendo dal primo strato si elaborano le informazioni più semplici, fino ad arrivare a quelle più complesse. Quanti più livelli intermedi ci sono, tanto è più efficace il risultato, ma la scalabilità è strettamente correlata al dataset, ai modelli matematici e alle risorse computazionali. Seppur la grande richiesta di capacità computazionali possa rappresentare un limite, la scalabilità del deep learning, grazie all'aumento dei dati disponibili e degli algoritmi, è ciò che lo differenzia dal machine learning. Infatti, mentre i sistemi di deep learning migliorano le proprie prestazioni all'aumentare dei dati, i sistemi di machine learning, una volta raggiunto un certo livello di performance, non sono più scalabili, nemmeno aggiungendo esempi e dati di training alla rete neurale. Questi sistemi sono difficili da

addestrare a causa del numero di strati della rete che può rendere difficile calcolare le regolazioni che devono essere apportate in ogni fase del processo di addestramento; questo perché si usano gli algoritmi di retropropagazione dell'errore (backpropagation) attraverso i quali si rivedono i pesi della rete (le connessioni tra i neuroni) in caso di errori.

Deep Learning nel mondo reale

Esistono vari framework per l'apprendimento automatico, ognuno dei quali è più o meno specifico per determinate applicazioni, un esempio è TensorFlow: una libreria open source supportata da Google che fornisce moduli testati e ottimizzati per la realizzazione di algoritmi da impiegare in diversi tipi di software e con diversi linguaggi di programmazione. Un altro framework è PyTorch, supportato da Meta (Facebook).

Alcuni casi d'uso che utilizzano queste tecnologie sono: la computer vision per le auto a guida autonoma (Tesla Autopilot basato su PyTorch), robot droni impiegati per la consegna di pacchi o assistenza in casi di emergenza, riconoscimento e sintesi vocali per chatbot e robot, riconoscimento facciale per la sorveglianza, riconoscimento di immagini a scopi medici, individuazione di sequenze genetiche e sistemi di analisi per la manutenzione predittiva su un'infrastruttura.

IA per gli attaccanti

Come l'intelligenza artificiale può aiutare a compensare la mancanza dei team di sicurezza e a rilevare anomalie nell'elaborazione di grandi quantità di dati, gli attori delle minacce possono armare l'IA per automatizzare azioni come la selezione del bersaglio o il tempismo dell'attacco per evitarne il rilevamento. Deepfake, impersonificazione umana e scoperta di password tramite l'intelligenza artificiale sono alcuni scopi malevoli; l'uso improprio dell'IA è una tendenza destinata a crescere. Ad esempio, in passato i criminali informatici sono stati in grado di copiare e manipolare il modello di machine learning per Proofpoint Email Protection per consentire alle e-mail dannose di passare attraverso i filtri.

5.1 IA per il rilevamento di attacchi informatici

Negli ultimi anni gli attacchi informatici sono diventati molto articolati e raramente vengono effettuati da un singolo computer. La maggioranza di essi è effettuata da persone che gestiscono reti di computer (bot) controllati da software che automatizzano la rilevazione della vulnerabilità e attuano l'attacco, rendendo tutto più semplice. Per questo motivo non si può pensare di far mitigare gli attacchi a singole persone, ma servono sistemi in grado di competere con la potenza di calcolo dei software automatizzati e stare al passo con i criminali informatici.

In questo viene in aiuto l'intelligenza artificiale che riesce a fornire un'intelligenza predittiva superiore leggendo il contenuto di articoli, notizie e studi recuperati automaticamente riferiti alle minacce informatiche.

L'IA rende possibile identificare comportamenti dannosi che non si sono mai visti prima, basandosi su due ipotesi:

- ciò che è anomalo è dannoso;
- ciò che è dannoso genera un'anomalia.

Queste due assunzioni non sempre si verificano, dando luogo a falsi positivi e falsi negativi.

I vecchi sistemi per il rilevamento si basavano su regole che una volta impostate andavano cambiate manualmente. Gli attaccanti motivati sono adattabili: cercano di capire come sono configurati i sistemi di rilevamento per aggirarli. Per questo viene in aiuto l'intelligenza artificiale che si adatta alle nuove situazioni e varia dinamicamente i parametri di rilevamento. In questo modo è possibile controllare una grande quantità di dati mantenendo basso il numero di errori.

Da molti anni i motori di analisi euristica e di riconoscimento dei pattern d'azione dei sistemi operano con tali tecnologie per aumentare la protezione e fornire una difesa innovativa in grado di adattarsi alle tante minacce presenti in ambito informatico. Un esempio di sistema efficace attivo nella protezione è Watson for Security, realizzato da IBM. Lo strumento permette agli analisti umani di avere una

piattaforma su cui lavorare mitigando i rischi delle minacce. Il prodotto effettua un'analisi approfondita di tutti i dati disponibili fornendo all'utilizzatore un rapporto completo sulle attività e su quanto è accaduto all'interno del sistema analizzato. In questo caso si hanno tutte le informazioni, ma spetta alla persona trarre le conclusioni e decidere le azioni da intraprendere. Watson for Security non è l'unico software relativo alla sicurezza informatica a sfruttare l'intelligenza artificiale, ma è differente da altri programmi come Darktrace, che usano comunque l'AI. In questo caso viene analizzato il traffico da cui si apprendono i normali flussi. L'intelligenza artificiale entra in gioco per scovare anomalie all'interno di tutti i dati raccolti ed esaminati. Si rivela un ottimo alleato quando si installano automaticamente software scaricati da internet e questi entrano in funzione eseguendo operazioni sospette. Molto utile, inoltre, in ambito di protezione dei dati aziendali, soggetti ad attacchi di spionaggio industriale.

L'intelligenza artificiale, nell'ambito della sicurezza informatica, non si occupa solamente di individuare le minacce e di indagare sulle attività all'interno della rete, ma opera anche attraverso la prevenzione evitando le intrusioni non autorizzate e proteggendo i dati sensibili contenuti nei dispositivi. L'IA agisce memorizzando i comportamenti degli utenti sul web, il modo con cui digitano le password e le informazioni scambiate così da avere un quadro della situazione e attuare le misure necessarie per garantire la sicurezza. Il software è capace di verificare se chi sta visitando un sito internet è una persona oppure un robot, quindi prende in esame il modo con cui vengono cliccati i tasti del mouse, la velocità di movimento, la durata dei click e tutto ciò che permette di monitorare i visitatori delle pagine web.

6 IBM QRadar Security Intelligence Platform

SIEM

Un SIEM, acronimo di Security Information and Event Management, è una soluzione essenziale per la sicurezza aziendale. Aiuta i responsabili della sicurezza a evidenziare le aree che richiedono un intervento attraverso misure correttive o migliorative. Il SIEM è una combinazione di due sistemi: SIM (Security Information Management), che gestisce la raccolta e la gestione dei log non in tempo reale, e SEM (Security Event Management), che esegue il monitoraggio e la gestione degli eventi che si verificano internamente fornendo la correlazione e l'aggregazione in tempo reale. Combinando questi due sistemi, i log raccolti possono essere analizzati per evidenziare eventi o comportamenti di interesse, rilevando così attività anomale.

La ricezione dei log può provenire da molte fonti:

- strumenti di sicurezza: IDS (Intrusion Detection System), IPS (Intrusion Prevention Systems), honeypots, firewall;
- dispositivi di rete: router, switch, server DNS, access point;
- apparati: dispositivi degli utenti, server di autenticazione, database.

Il potenziale di un SIEM risiede nell'aggregare dati vitali da più fonti, visualizzare analisi e correlazioni in tempo reale per identificare comportamenti anomali, segnali critici e generare avvisi. L'ultima generazione di SIEM 4.0 può includere sistemi euristici in grado di identificare vari tipi di attacchi informatici, come attacchi 0-day, attacchi DDoS e attacchi di forza bruta. In questo modo, è possibile rilevare attività anomale attuando le politiche di sicurezza stabilite dall'organizzazione per determinare quali azioni devono essere intraprese. Infine, è possibile avviare azioni di risposta automatizzata agli attacchi per bloccare o rimuovere il traffico potenzialmente dannoso o ridurre le prestazioni ristabilendo l’operatività dell'infrastruttura IT.

Un esempio di SIEM 4.0 è QRadar di IBM.

Uno dei parametri per il corretto funzionamento di un SIEM è la scelta delle informazioni che deve raccogliere da ogni singolo componente del sistema da

proteggere, pertanto è necessario definire perimetri e obiettivi, poiché sarebbe impossibile monitorare l'intero sistema. Il SIEM migliora il tempo medio di rilevamento (MTTD) e il tempo medio di risposta (MTTR) eliminando i flussi di lavoro manuali associati all'analisi approfondita degli incidenti di sicurezza.

Le soluzioni SIEM sono una scelta comune per le organizzazioni soggette a varie forme di conformità normativa. Grazie alla raccolta e all'analisi automatizzate dei dati che fornisce, il SIEM è uno strumento prezioso per la raccolta e la convalida dei dati in un'infrastruttura aziendale. È possibile generare report di conformità in tempo reale per PCI-DSS, GDPR, HIPPA, SOX e altri standard, riducendo l'onere della gestione della sicurezza e rilevando, tempestivamente, potenziali violazioni in modo che possano essere affrontate. Molte soluzioni SIEM sono dotate di plug-in predefiniti e pronti all'uso che generano report automatici progettati per soddisfare i requisiti.

QRadar

Oggi la sicurezza aziendale è predittiva: analizzando il comportamento degli utenti e dei sistemi che accedono a una rete aziendale, si costruisce uno scenario per prevedere e prevenire il prossimo attacco. L'ambito tecnico che si occupa dell'analisi delle informazioni a protezione delle reti aziendali è definito SIEM e utilizza strumenti specifici. Una delle più efficaci, considerata da diversi analisti, è IBM QRadar Security Intelligence Platform. Si tratta di un'architettura unificata per l'analisi di eventi di log, di flussi di rete, di pacchetti, delle vulnerabilità, dei dati relativi alle risorse aziendali e delle violazioni di sicurezza. In altre parole, l'architettura sorveglia il controllo degli accessi alla rete e ai sistemi aziendali e li monitora in tempo reale. Così, grazie alle integrazioni con altre soluzioni IBM, i comportamenti rilevati possono essere confrontati con un archivio costantemente aggiornato per l'immediata identificazione di attività sospette.

Con IBM QRadar Security Intelligence, non solo vengono monitorate le attività e si segnalano quelle a rischio, ma è anche possibile ripercorrere a ritroso il percorso degli attaccanti, e intervenire immediatamente. Inoltre, la soluzione di IBM identifica i problemi di configurazione di rete e dei dispositivi connessi e può essere molto utile nel soddisfare i criteri di conformità sul trattamento dei dati previsti dalla normativa.

Il SIEM QRadar consente di vedere se una determinata comunicazione è avvenuta, se è stato eseguito malware, da chi, dove e se è avvenuto l'accesso alle risorse aziendali. È inoltre possibile ottenere un profilo di rischio per ciascun utente e identificare eventuali attacchi interni, tentativi di escalation dei privilegi o violazioni di dati riservati. L'analisi di QRadar e gli algoritmi associati consentono di ridurre al minimo i falsi positivi. Questi eventi sono raccolti in un database e le ricerche sono possibili attraverso un'interfaccia centralizzata. Grazie alla possibilità di aggregare i dati, è possibile ottenere visualizzazioni in base ai propri interessi, dagli utenti executive interessati a diagrammi e thread fino alle strutture operative interessate ai dettagli, riducendo i tempi di elaborazione delle informazioni.

6.1 Componenti di QRadar

Per garantire la scalabilità del sistema e gestire i dati distribuiti, QRadar è costituito da diversi componenti. Le diverse distribuzioni possono includere:

Console: la console fornisce l'interfaccia utente e consente la visualizzazione in tempo reale di eventi e flussi. Consente inoltre di generare report, visualizzare lo stato delle risorse, delle Offense e gestire parametri amministrativi.

Event Collector: l'Event Collector è un componente che raccoglie eventi da origini log locali e remote e normalizza i dati grezzi, formattandoli per l'analisi del software. A questo componente viene assegnata una licenza EPS (Events Per Second) e ha la capacità di raggruppare e unire eventi identici per ridurre il consumo di risorse. L'Event Collector non salva gli eventi localmente, ma li raccoglie ed elabora prima di inviarli all'Event Processor che li memorizzerà; inoltre, può utilizzare limitatori di banda per programmare la trasmissione degli eventi su una WAN.

Event Processor: l'Event Processor utilizza un Custom Rules Engine (CRE) per elaborare gli eventi raccolti da uno o più Event Collector. Quando gli eventi sono associati a una regola, l'Event Processor eseguirà l'azione definita in risposta ad essa, dopo averli ricevuti. Ciascuno di questi componenti ha una memoria locale

in grado di contenere eventi; opzionalmente, è possibile inserire questi eventi in un Data Node (un nodo dedicato a questo scopo). La velocità di elaborazione degli eventi è determinata dalla licenza EPS; se questa soglia viene superata, gli eventi vengono archiviati in un buffer e vi rimangono fino a quando il numero di eventi ricevuti al secondo diminuisce. Tuttavia, se ciò non accade, la coda potrebbe saturarsi e il sistema procederà a scartare gli eventi e avvisare l'utente.

Data Node: i Data Node consentono alle nuove implementazioni QRadar di aumentare la potenza di archiviazione e di elaborazione. Questi componenti migliorano notevolmente la velocità di ricerca fornendo più risorse hardware per le query.

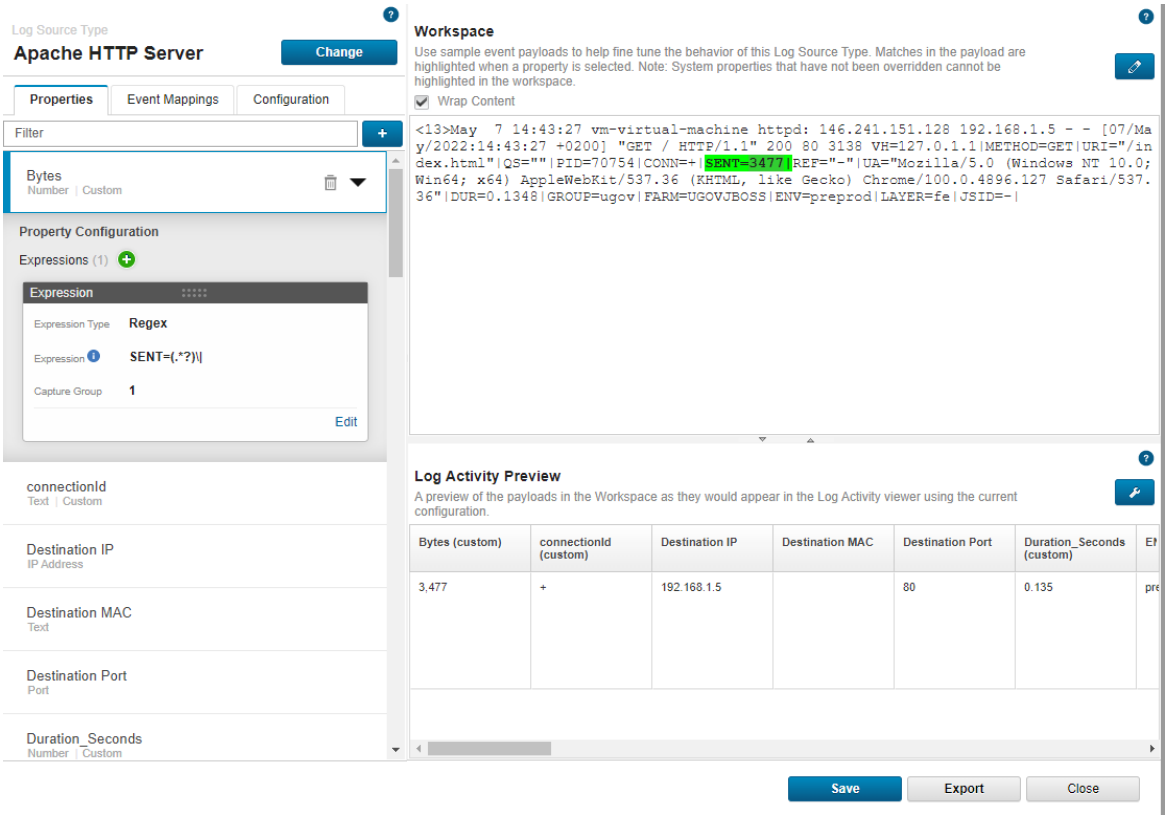
App Host: poiché QRadar è un software in grado di ospitare altre applicazioni e plug-in, è possibile utilizzare un App Host dedicato per eseguire queste estensioni senza influire sulla potenza di elaborazione della Console. Infatti, gli App Host forniscono ulteriori risorse di archiviazione, memoria e CPU che sono fondamentali per l'esecuzione di applicazioni particolarmente onerose.

Per garantire l'integrità dei dati raccolti, è disponibile la possibilità di generare file di hash. Questa funzionalità, che deve essere abilitata, consente di generare un hash per qualsiasi sistema che si occupa di scrivere dati di eventi e flussi, scrivendo gli hash in memoria prima che i file degli eventi vengano scritti su disco. In questo modo, un file di log non può essere manomesso prima che venga generato il file hash e la sua integrità può essere verificata in qualsiasi momento. QRadar supporta gli algoritmi MD e SHA.

La piattaforma di IBM è dotata anche di un Device Support Module (DSM), ovvero un modulo di codice che analizza gli eventi ricevuti da più origini log e li converte in un formato tassonomico standard che può essere visualizzato come output. Esiste un DSM corrispondente per ogni tipo di origine log. Ad esempio, il DSM IBM Fiberlink MaaS360 analizza e normalizza gli eventi dalle origini log IBM Fiberlink MaaS360. Dopo la raccolta degli eventi e prima che possa iniziare la correlazione, deve essere eseguita un'appropriata normalizzazione degli eventi dei singoli dispositivi. Per normalizzazione si intende il processo di mappatura delle

informazioni su nomi di campi comuni, come nomi di eventi, indirizzi IP, protocolli e porte.

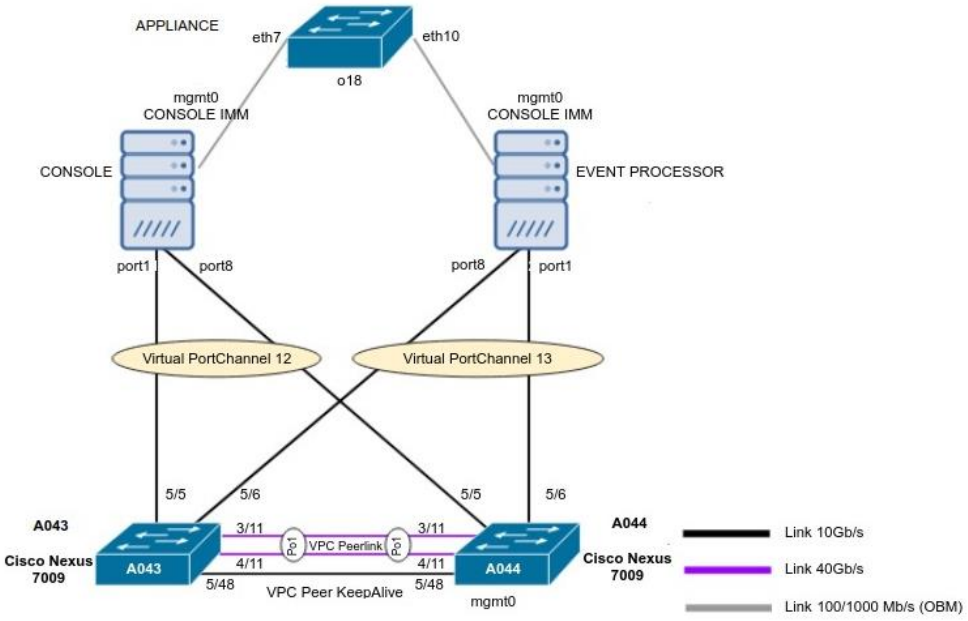
Se una rete aziendale dispone di uno o più dispositivi di rete o di sicurezza per i quali QRadar non fornisce un DSM corrispondente, è possibile utilizzare un tipo di origine log personalizzato. QRadar può integrarsi con la maggior parte dei dispositivi e qualsiasi origine di protocollo utilizzando tipi di origine di log personalizzati. Di seguito viene mostrato un esempio di configurazione DSM per la sorgente di log Apache HTTP Server.



Per impostazione predefinita, ogni DSM dispone di campi già preimpostati per interpretare i log standard, tuttavia è possibile definire proprietà personalizzate per integrare le proprietà di base configurando appropriate espressioni regolari. È possibile anche decidere se i campi aggiuntivi devono far parte del database in modo che possano essere ricercabili tramite query AQL o specifiche regole.

Nell'ambiente QRadar studiato in questo progetto, installato presso il CINECA di Casalecchio di Reno (BO), sono state effettuate alcune stime per valutare l'occupazione dello storage nel tempo. Attualmente CINECA riceve una media compresa tra 10.000 a 20.000 eventi al secondo, occupando a 1,3 TB per i record e 575 GB per il payload (analisi effettuata nei primi 15 giorni di novembre 2021). Il record rappresenta un evento normalizzato e il payload si riferisce al contenuto grezzo della richiesta associata al record. A titolo di stima, si può notare che tra payload e record, in media, vengono utilizzati 125 GB al giorno, ovvero circa 3,8 TB al mese.

Per quanto riguarda le specifiche tecniche del sistema di produzione, l'installazione consiste in due dispositivi fisici, uno avente ruolo di Console e l'altro di Event Processor, ciascuno con 256 GB di RAM, 48 core e 70 TB di disco.



La licenza in uso consente di gestire fino a 15.000 eventi per secondo come SIEM, tuttavia le appliance riescono a sopportare un carico fino a 40.000 EPS (15.000 possono essere passate al SIEM, le restanti conservate per l'archiviazione).

6.2 Interrogazione dei Log tramite AQL

Ariel Query Language (AQL) è un linguaggio strutturato utilizzato per comunicare con il database Ariel di cui è dotato QRadar. AQL viene utilizzato per interrogare e manipolare dati di eventi e flussi presenti nel database; consente di ottenere informazioni che normalmente non possono essere visualizzate sui grafici della Console. La struttura di questo linguaggio è molto simile a SQL e può essere riassunta come segue:

```
AQL Structure
[SELECT *, column_name, column_name]
[FROM table_name]
[WHERE search clauses]
[GROUP BY column_reference*]
[HAVING clause]
[ORDER BY column_reference*]
[LIMIT numeric_value]
[TIMEFRAME]
```

Il database QRadar è composto da due tabelle "events" e "flows", i suoi campi di database sono principalmente quelli indicati nel DSM delle varie sorgenti di log e possono essere estratti in qualsiasi momento tramite query. Per effettuare ricerche più approfondite e specifiche sulla grande quantità di dati gestiti da QRadar, è possibile aggiungere estensioni in linguaggio JavaScript. Queste consentono di definire funzioni che possono essere richiamate direttamente nello statement AQL. Di seguito viene mostrata la struttura del file XML che consente l'installazione e la definizione delle estensioni.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<content>
    <custom_function>
        <namespace>nsesempio</namespace>
        <name>esempio</name>
        <return_type>Boolean</return_type>
        <parameter_types>String</parameter_types>
        <execute_function_name>execute</execute_function_name>
```

```
        <script_engine>javascript</script_engine>
        <script>
        function execute(raw_payload) {
             if (raw_payload.length==90) {
                  return false;
             }
             return true;
        }
</script>
        <username>admin</username>
        <author>John Doe</author>
    </custom_function>
</content>
```

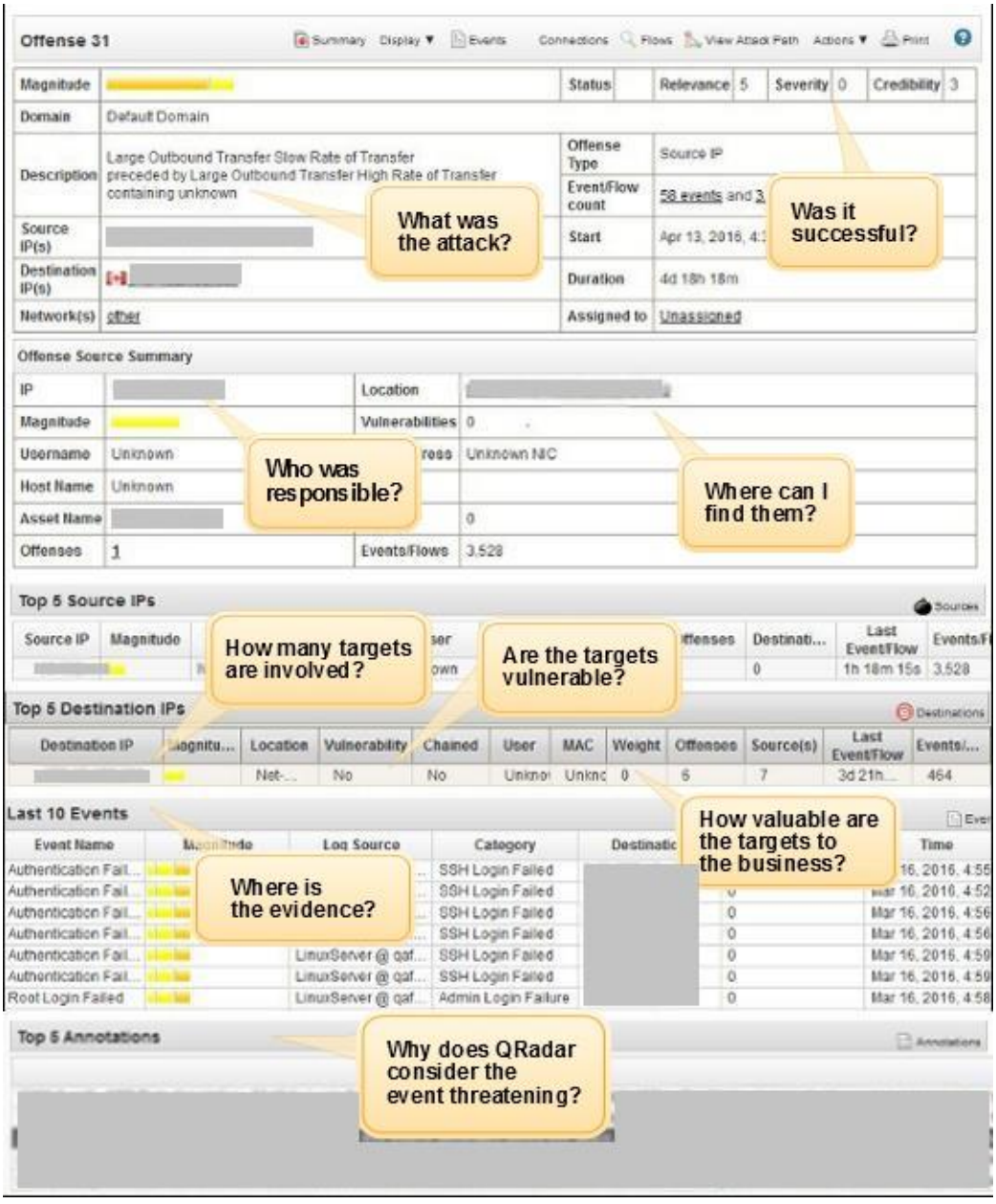
Come si può notare dal codice esposto, è necessario definire il nome della funzione, il relativo namespace, il tipo di variabili di ritorno, il tipo di variabili in ingresso e il nome della funzione JavaScript da invocare; infine vengono indicati altri parametri informativi come l'autore e l'username. Una volta scritto il file .xml è necessario compattarlo in un file zip e installarlo tramite l'interfaccia di QRadar. A questo punto è possibile richiamare la funzione personalizzata in uno statement AQL come segue:

```
SELECT * FROM Events WHERE NSESEMPIO::ESEMPIO(UTF8(payload)) LAST "5 minutes"
```

È importante notare che la struttura dello storage di QRadar è suddivisa in directory del tipo ANNO/MESE/GIORNO/ORA/MINUTO e il consolidamento dei file viene eseguito minuto per minuto. Questo comporta l'impossibilità di includere nei risultati di una query i log ricevuti nel minuto in corso, in quanto il file risulterebbe ancora aperto e i log non consolidati. Supponendo infatti di ricevere un log di interesse alle ore 15:35:10, questi non sarà leggibile dalle query AQL fino alle ore 15:36:00.

6.3 Segnalazione di anomalie: Offense

QRadar utilizza le regole per monitorare eventi e flussi all'interno della rete e rilevare le minacce alla sicurezza. Quando un evento o un flusso soddisfa i criteri definiti nella regola, il sistema genera una Offense per attirare l'attenzione su un possibile attacco o violazione delle policy di sicurezza. Al fine di determinare la causa e il responsabile, il SIEM fornisce una finestra di riepilogo che riassume tutte le informazioni sul contesto e sull'accaduto.

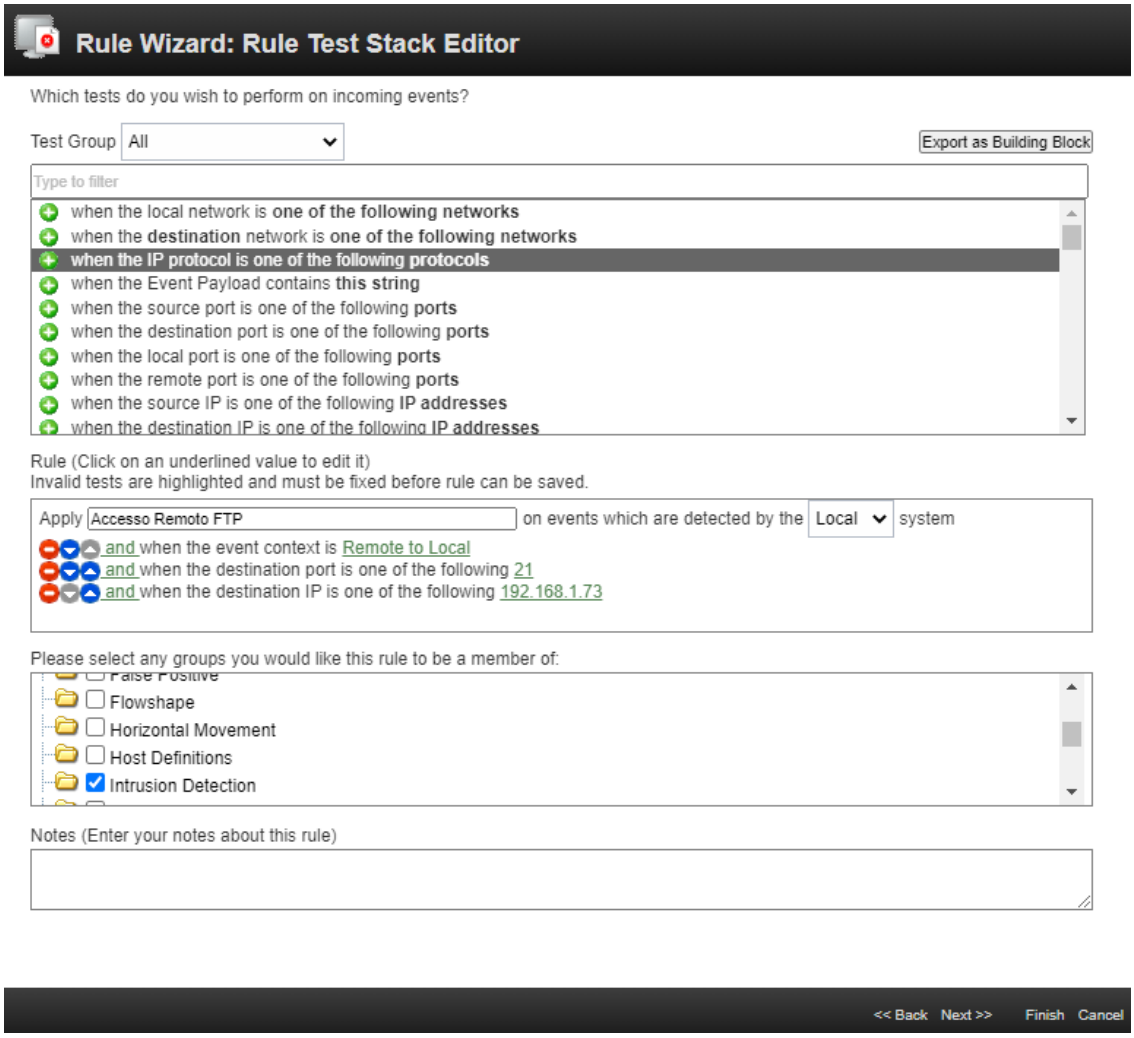


Per generare un’Offense, è necessario definire alcune regole nel Custom Rules Engine (CRE) che forniscano informazioni su come sono raggruppate, i tipi di criteri che utilizzano e le risposte che ogni regola deve generare. Il CRE visualizza le regole e i BuildingBlock utilizzati da QRadar e li memorizza in due elenchi separati.

Un BuildingBlock è un componente logico che utilizza gli stessi criteri che possono essere inseriti in una regola, ma a differenza delle regole, non esegue alcuna azione in risposta a un evento che si verifica. Sono spesso usati per comporre regole più complesse o per creare raggruppamenti comuni. Ad esempio, è possibile creare un BuildingBlock per rappresentare l'insieme IP di tutti i server di posta nella rete, quindi utilizzarlo in altre regole per escludere quegli host. Nel caso di modifica del set IP, è sufficiente modificare solo il blocco senza dover riscrivere tutte le regole che lo utilizzano. QRadar fornisce una serie di BuildingBlock comunemente usati per impostazione predefinita, ma è possibile definirne di personalizzati.

Una regola è un insieme di condizioni che, se soddisfatte, attivano un'azione specifica. Ciascuna di esse viene creata tramite l'editor delle regole e può essere configurata per acquisire e rispondere a eventi specifici, sequenze di eventi o attacchi. QRadar fornisce una serie di criteri configurabili che possono essere combinati tra loro per creare nuove regole, così come fornisce una serie di azioni in risposta a eventi come l'invio di e-mail o la generazione di messaggi di log.

Di seguito sono riportati alcuni criteri che possono essere impostati per rilevare gli attacchi remoti alle risorse FTP, generalmente accedute dalla rete locale.



6.4 API

QRadar fornisce API RESTful che consentono di inviare richieste HTTPS a URL specifici della console. Per fare ciò, è sufficiente utilizzare l'implementazione HTTP di qualsiasi linguaggio di programmazione che consenta di inviare tali richieste. In ogni richiesta devono essere trasmesse le informazioni di autenticazione e i parametri identificativi della richiesta. Le informazioni di autenticazione devono essere incluse nell'intestazione HTTP e riguardano un nome utente e una password o un token di autenticazione. Il primo metodo è sconsigliato in quanto potrebbe comportare rischi per la sicurezza; è sempre meglio utilizzare un token che può essere generato dalla console QRadar e può essere associato ad un ruolo

e profilo di sicurezza specifici. Inoltre, quest'ultimo consente di specificare una data di scadenza dopo la quale, il token, non sarà più utilizzabile.

Quando si invia una richiesta API, il server restituisce una risposta HTTP con il codice di stato e i dettagli della risposta (se presenti), generalmente in formato JSON. Tramite il codice di stato è possibile sapere se la richiesta è andata a buon fine, così da procedere con l'estrazione dei dati contenuti nel corpo.

Grazie alle API e alle altre funzionalità messe a disposizione di QRadar, sarebbe infatti possibile controllare un firewall in maniera tale da mitigare gli attacchi. Riprendendo l'esempio di accesso remoto FTP del paragrafo precedente, si potrebbe pensare di sfruttare le API per accedere all'offesa generata, recuperare l'indirizzo IP dell'attaccante e, in seguito, trasmettere questo dato ad un firewall che si occuperà di inserirlo in una blacklist, quindi limitare le conseguenze. Queste sono alcune delle potenzialità fornite da QRadar.

7 Analisi dei comportamenti per rilevare anomalie

In questa tesi ci si è soffermati sui problemi che possono scaturire dall'utilizzo delle sessioni. Avendo un server in grado di supportare un numero di sessioni limitato ci si è posti il dubbio se, nel caso in cui si vadano a occupare tutte le sessioni disponibili, i nuovi utenti potessero ancora accedere al sito.

Sapendo che le sessioni vengono generate ogni volta che un utente entra nel sito senza sessione o con una non valida, si è pensato che in questo modo fosse possibile per un attaccante rendere il sito non accessibile inviando un gran numero di richieste in breve tempo. Queste richieste andranno a consumare le sessioni in maniera molto rapida; inoltre, rispetto alle risorse richieste per realizzare un attacco DoS o DDoS, effettuare un attacco di questo tipo è molto semplice.

Nel caso analizzato, sulla piattaforma QRadar si avevano a disposizione i log generati dalle applicazioni di interesse. Utilizzando queste informazioni era necessario individuare i comportamenti dannosi che potessero causare l'esaurimento delle sessioni e l'indisponibilità del servizio. Il primo passo da affrontare per rilevare le anomalie è stato quello di analisi, in cui si è reso necessario studiare i comportamenti degli utenti. In questo studio sono state prese in considerazione diverse informazioni come gli orari di punta in cui venivano utilizzati gli applicativi, il numero di pagine web visitate in un arco temporale e la tipologia di richieste effettuate. Bisogna inoltre notare che non tutte le interazioni sono eseguite da esseri umani, quindi alcune tipologie di richieste andrebbero escluse onde evitare di deviare le statistiche.

La gestione delle problematiche di questo tipo si articola in due fasi:

- rilevazione, ovvero ciò che si discosta in modo più o meno netto dalla normalità; il concetto di "normalità" non è semplice da definire e viene modellato in maniera diversa in base alle varie tecniche di rilevazione: per esempio, in questo lavoro di tesi l'enfasi è stata posta sul numero di sessioni generate in un periodo di tempo.
- gestione e mitigazione, la quale può essere effettuata manualmente o per mezzo di procedure informatiche automatizzate: alcuni applicativi registrano le

varie attività di rete sospette, in modo da trasmettere degli allarmi ai soggetti interessati alle analisi di rete; altri agiscono in modo più deciso, cercando autonomamente di mitigare o risolvere la situazione anomala che si è verificata. Ovviamente gli approcci sopra citati possono coesistere.

7.1 Utilità di una baseline

L'anomaly detection è un complesso sistema di monitoraggio degli andamenti di una metrica che, quando si verifica un'anomalia, permette di avere un alert appena accade. Per anomalia si intende una deviazione dal comportamento standard che potrebbe verificarsi sia in positivo che in negativo. A questo punto è importante distinguere tra anomalia e trend.

Un trend è un comportamento che si consolida nel tempo e diventa costante. In un grafico può essere rappresentato da una curva o una linea retta che vanno verso l'alto o verso il basso; non presenta particolari picchi e rimane sufficientemente costante nel tempo.

L'anomalia, invece è una variazione di un trend. Può essere rappresentata a livello grafico come un picco o una cuspide che permette di identificare qualcosa di completamente diverso da ciò che accade di solito.

Monitorare i trend e le anomalie è fondamentale per tutti i progetti, ma nell'ambito della sicurezza informatica assume una particolare rilevanza, dato che, dai comportamenti degli utenti, è possibile rilevare eventuali problematiche. Infatti, un traffico di grande importanza registrato in momenti in cui solitamente si sono rilevati pochi accessi potrebbe essere sintomo di un attacco DoS. Allo stesso modo, nel momento in cui in orari di punta si rileva un numero notevolmente più basso di interazioni potrebbe essere sintomo di un malfunzionamento al servizio.

In questo senso diventa fondamentale avere un sistema di anomaly detection, in grado di segnalare in tempo reale una deviazione dal trend e consentire di mettere in atto azioni di recupero. Senza un sistema di alerting è molto difficile scoprire queste cuspidi sui grafici e spesso si interviene in ritardo.

Per consentire di rilevare i comportamenti che si discostano dal trend è innanzitutto necessario studiare il trend e sviluppare una baseline in grado di rispecchiarlo. Bisogna inoltre considerare il fatto che il comportamento degli utenti può cambiare nel tempo, quindi una baseline sviluppata in un determinato periodo, molto probabilmente si dimostrerà errata in periodi successivi. Questo può essere causato da molti fattori tra cui la stagionalità del servizio, la modifica di alcune funzionalità degli applicativi, la modifica del bacino di utenza e l'incremento o la diminuzione degli utilizzatori. Prendendo ad esempio un software gestionale di un'università italiana è molto probabile che rilevi picchi di utenza nelle prime ore del mattino (8:00-10:00), momento in cui tutti gli impiegati iniziano a lavorare e si collegano al sistema. Allo stesso modo sarebbe anomalo ricevere una grande quantità di richieste in orari notturni, considerando che il software, come scritto, è destinato ad un pubblico italiano.

Qualora però la software house, in futuro, decidesse di rendere disponibile lo stesso applicativo ad un bacino di utenza residente negli Stati Uniti le abitudini saranno ovviamente diverse e la baseline analizzata per il pubblico italiano non potrà essere applicata a quello statunitense.

Un esempio analogo, si può applicare ai vari mesi dell'anno: sicuramente in Italia, nel mese di agosto ci si aspetta una diminuzione delle richieste. Queste considerazioni sono relativamente semplici da fare per un essere umano, tuttavia un computer non è in grado di comprendere tali situazioni. La realizzazione di una baseline consente alla macchina di eseguire una fase di training durante la quale può rilevare e apprendere in maniera automatica i comportamenti degli utenti. È compito del programmatore realizzare meccanismi in grado di generare una baseline il più possibile corretta per un determinato scopo e adattabile nel tempo. Per fare questo è necessario assegnare alcuni pesi alle variabili che vengono prese in considerazione al fine di ottenere un tuning dei parametri corretto.

Infine, è necessario precisare che, ad oggi, non esiste una soluzione specifica per modellare perfettamente una determinata situazione; si dovrà quindi considerare l'ipotesi che vengano generati falsi positivi e falsi negativi. Anche in questo caso

spetta al team di progettazione giungere al miglior compromesso.

7.2 Realizzazione di una baseline

La Web Application studiata in questo progetto di tesi trasmette a QRadar tutte le interazioni che l'utente svolge all'interno della pagina. Nello specifico, quando un utente si collega al sito web per la prima volta, il server della Web Application provvede ad assegnargli una sessione. Queste operazioni sono visibili all'interno del SIEM grazie al fatto che l'utente alla prima interazione, non trasmette alcuna sessione, quindi il relativo campo, denominato JSID, è valorizzato con un trattino '-'. A partire dall'interazione successiva il campo JSID viene popolato con l'ID di sessione generato dal server.

In condizioni normali quindi, a livello di log, ci si aspetta un singolo log con JSID='-' e i successivi con il JSID valorizzato. Il numero di log successivi è variabile a seconda di quante interazioni svolge l'utente nel periodo di validità della sessione o fino a quando non esegue il logout dal sistema.

È necessario sottolineare che la procedura sopra descritta deve aggregare i dati aventi gli stessi valori di SourceIP e VirtualHostName (il nome dell'applicazione). Si prende in considerazione il VirtualHostName e non l'indirizzo IP di destinazione, poiché un'applicazione può essere ospitata su più server fisici ed è compito del bilanciatore di carico indirizzare le richieste ad un server specifico. Grazie a questa aggregazione di risultati, infatti, è possibile valutare quante interazioni ha eseguito un determinato cliente verso uno specifico servizio offerto dall'infrastruttura. Le interazioni devono distinguersi in interazioni svolte senza sessione e interazioni svolte con la sessione. Partendo da questa premessa è normale aspettarsi un numero elevato di interazioni eseguite con sessione e un numero molto ridotto di richieste senza session ID.

Dato che le sessioni hanno una durata temporale limitata (solitamente di alcuni minuti) è del tutto normale che un utente faccia uso di più sessioni all'interno di un arco temporale ampio. È infatti possibile che un utente si sia assentato dalla

propria postazione per un tempo superiore alla durata della sessione e in questo caso il server abbia proceduto all'invalidazione. Al suo ritorno, il cliente, tenterà di collegarsi alla pagina web trasmettendo l'ID di sessione di cui era in possesso precedentemente; a questo punto il server constaterà che quella sessione non è più valida e procederà a generarne una nuova.

In questo caso a livello di log non verrà più mostrato il campo JSID='-' poiché di fatto il cliente non si è collegato per la prima volta al sito web, ma ha tentato di proseguire la navigazione con una sessione scaduta. Come si evince, non è quindi più possibile basare l'analisi delle anomalie solo sul numero di trattini rapportati alle sessioni valide ma è necessario attuare sistemi più articolati.

Per risolvere questo problema si potrebbe pensare di contare il numero di sessioni utilizzate da parte di uno stesso cliente all'interno di un arco temporale e realizzare una media di sessioni che utilizza di norma un utente, abbandonando il conteggio dei trattini.

Mettendosi dalla parte di un attaccante però si dimostra che questa soluzione non è sufficiente. Un attaccante che ha lo scopo di rendere un servizio indisponibile saturando le sessioni applicative, infatti, può agire in due modi: accedere senza sessioni oppure accedere con sessioni non valide. In entrambi i casi, come visto nel paragrafo dedicato alla gestione delle sessioni, il server risponde con la creazione di una nuova sessione che sottrae un posto ad un reale utilizzatore.

Dato che la Web Application analizzata non trasmetteva un log contenente l'avviso di creazione di una sessione, si potevano verificare i casi descritti di seguito.

Qualora l'attaccante esegua l'accesso all'applicazione senza sessione, il server web trasmette a QRadar un log contenente il campo JSID='-' poiché la richiesta ricevuta non presentava alcun ID di sessione. A questo punto il server risponde all'attaccante comunicando il SessionID, ma quest'ultimo scarta la risposta che non sarà loggata dal servizio. Ripetendo più volte questa operazione, all'interno del SIEM si noterà un numero elevato di JSID='-' per una determinata coppia SourceIP, VirtualHostName e nessun ID di sessione.

Qualora l'attaccante decidesse di eseguire l'accesso all'applicazione con una sessione inventata o non valida, in maniera analoga al caso precedente, sul SIEM si visualizzerà un numero elevato di log contenenti un SessionID di cui non si è certi della reale esistenza e nessun log contenente il valore JSID='-'.

Analizzando i due casi descritti si può pensare di verificare la quantità di trattini rilevati e la quantità di sessioni distinte trasmesse. Considerando infatti il numero distinto di SessionID è possibile calcolare la differenza tra questi due valori, che in condizioni ideali deve essere pari a 0. Dato che nell'ambiente reale possono accadere diverse situazioni non determinabili a priori è comunque normale avere una differenza diversa da zero; questo perché l'analisi, che viene fatta a posteriori, deve prendere in considerazione un periodo temporale all'interno del quale non è detto che sia stata generata la sessione. Infatti, la prima interazione dell'utente (con JSID='-') potrebbe essere stata eseguita poco prima del periodo preso in esame e quindi non essere rilevata, causando una differenza positiva o negativa.

Ulteriormente, un attaccante particolarmente ferrato in materia potrebbe pensare di ingannare questo sistema trasmettendo in maniera alternata una richiesta senza sessione e una con ID di fantasia; in tal caso la differenza sarebbe pari a zero e non farebbe scattare nessun allarme.

Alla luce di quanto esposto, è quindi necessario affinare ulteriormente l'analisi del traffico considerando un numero di sessioni massimo che può generare ogni singolo utente verso una specifica applicazione web. Questo parametro deve tenere in considerazione che alcuni gruppi di utenti operano dietro una rete provvista di NAT quindi è assolutamente normale visualizzare un numero molto elevato di sessioni provenienti da uno stesso indirizzo IP; si pensi, ad esempio, agli studenti che si collegano al sito web istituzionale tramite il WiFi dell'università.

Per considerare tutte le casistiche sopra elencate è stato realizzato uno script Python che sfruttando le API RESTful messe a disposizione da QRadar recupera la quantità di sessioni e trattini per ogni coppia SourceIP, VirtualHostName in un determinato arco di tempo. Una volta ottenuti questi due valori viene calcolato il massimo tra i due e il valore assoluto della differenza. Si è scelto di prendere in

considerazione il massimo tra sessioni e trattini poiché essi possono rappresentare in maniera più fedele il numero di sessioni effettivamente generate dal server.

A questo punto, per ogni coppia SourceIP, VirtualHostName, si ha il valore presunto di sessioni generate e il delta tra sessioni e trattini. Sulla base di questi valori è possibile produrre una baseline in grado di apprendere i comportamenti abituali. Questo script è programmato per eseguire con periodicità giornaliera e prendere in considerazione tutti gli eventi del giorno precedente. Durante l'analisi dei log divide la giornata in due periodi: giorno e notte. Per ognuno di questi periodi considera la baseline prodotta dal giorno precedente ed esegue una media pesata al fine di adattarsi ai nuovi valori rilevati.

È inoltre presente una soglia inferiore di default che, qualora raggiunta, comporta la rimozione del record dalla baseline. Questo consente di contenere il più possibile le dimensioni della baseline.

7.3 Rilevazione dell'esaurimento di risorse tramite QRadar

Dal momento in cui si ha a disposizione una baseline contenente i comportamenti ideali degli utenti è possibile utilizzarla per rilevare le anomalie, ovvero eventi che si discostano dal trend. Per realizzare questa operazione è stato sviluppato uno script Python in grado di leggere i dati forniti dallo script precedente e, interrogando il database di QRadar tramite API RESTful, verificare i comportamenti.

In primo luogo è necessario distinguere la fascia oraria in cui ci si trova, questo perché in base all'orario dei dati presi in considerazione sarà necessario utilizzare una baseline diversa: giorno o notte. Una volta appurato a quale fascia oraria si appartiene, si procede al recupero dei log di interesse da QRadar, dopodiché effettuando operazioni di outer join è necessario affiancare ad ogni log il valore soglia ad esso relativo, recuperato dalla baseline.

Durante questa procedura è necessario tenere conto dei periodi che vengono presi

in considerazione: i valori contenuti nella baseline, di default, sono calcolati sulla base di 14 ore per il giorno (8:00-22:00) e 10 ore per la notte; supponendo che lo script di analisi venga eseguito ogni 5 minuti è necessario riparametrare tali valori. Per questo motivo, quando lo script di analisi viene eseguito, il valore che sarà affiancato ai log recuperati sarà quello riparametrato.

Supponendo di avere una coppia SourceIP = 192.168.1.5, VirtualHostName = webapp1.cineca.it con valore 290 sulla baseline: si recuperano le sessioni e i trattini rilevati per questa coppia negli ultimi 5 minuti, quindi si calcola il 30% del valore contenuto nella baseline e si affianca alla riga. A questo punto tramite una query si verifica se il numero di sessioni o trattini è superiore al valore affiancato (in questo caso 87); in caso positivo dovrà essere inviata una Offense.

In alcuni casi il valore della baseline potrebbe risultare relativamente basso e rapportandolo al 30% si rischierebbe di ottenere un numero ancora più piccolo che potrebbe generare un numero molto elevato di Offense, di fatto falsi positivi. Per risolvere questa problematica è stata introdotta una soglia al di sotto della quale non è possibile scendere; in tal modo se il valore calcolato come 30% della baseline risultasse troppo piccolo verrebbe sostituito da questa soglia.

Quello appena esposto è un singolo metodo per rilevare le anomalie, tuttavia come anticipato nella sezione precedente è necessario prendere in considerazione anche la differenza fra trattini e sessioni, poiché una differenza importante potrebbe rappresentare un comportamento anomalo da parte di alcuni utenti. A questo scopo, oltre alla rilevazione già esposta viene valutato anche il valore di differenza che, qualora superi una soglia dedicata, causerà la generazione di una Offense. Entrando nello specifico, a seguito delle analisi di comportamento si è visto che una differenza superiore al 35% del massimo tra sessioni e trattini è anomala. Si è reso necessario utilizzare una percentuale poiché su un numero molto elevato di sessioni o trattini è normale avere una differenza più elevata rispetto al caso in cui le sessioni siano in numero ridotto. Anche in questa situazione è stata fissata una soglia inferiore per risolvere il caso in cui il calcolo della percentuale potesse dare come risultato un numero molto piccolo, con la

conseguenza di generare falsi positivi.

L'intero progetto è stato sviluppato cercando di ridurre al minimo i casi di falsi negativi, in quanto ritenuti più impattanti per l'organizzazione.

7.4 Funzionamento specifico

Il progetto è composto da due script Python, uno dedicato alla generazione della baseline e uno alla rilevazione delle anomalie. Il primo è stato progettato per eseguire una volta al giorno e mantenere sempre aggiornata la baseline a seconda del comportamento degli utenti. Si consiglia di eseguire questo programma nelle prime ore della notte, in maniera tale da ottenere al più presto i valori aggiornati su cui basare la successiva analisi.

Il secondo script, dedicato alla rilevazione delle anomalie è stato progettato per eseguire di default ogni 5 minuti e deve avere a disposizione una baseline calcolata dal programma precedente. A differenza del primo, questi mette a disposizione dell'utente alcuni parametri configurabili, descritti di seguito:

- -o consente di visualizzare le Offense direttamente sul terminale, senza trasmetterle a QRadar
- -v stampa i dettagli dell'esecuzione;
- -min <int> consente di modificare la periodicità di esecuzione, ad esempio per eseguire ogni 10 minuti. In questo caso sarà necessario eseguire il tuning dei parametri descritti di seguito. (Default: 5);
- -MS <int> consente di indicare il massimo numero di sessioni o trattini consentiti per le coppie SourceIP, VirtualHostName non presenti nella baseline;
- -DT <int> consente di indicare la soglia del valore di differenza fra trattini e sessioni, al di sopra della quale generare le Offense;
- -DP <int> consente di modificare la percentuale di differenza fra trattini e sessioni, al di sopra della quale generare le Offense (Default: 35).

- -BP <int> consente di modificare la percentuale da utilizzare per rapportare i valori contenuti nella baseline a quelli da prendere in considerazione nei minuti di esecuzione indicati con -min. (Default: 30).

Al fine di poter monitorare direttamente dalla Console di QRadar i risultati dell'analisi condotta e visto che tramite API non è possibile creare una nuova Offense, lo script si occupa di trasmettere a QRadar una tipologia di log creata ad hoc che causerà l'apertura dell'Offense contenente tutti i dettagli. Grazie a questa funzionalità l'utente di QRadar non dovrà preoccuparsi di eseguire gli script manualmente. A tal proposito la soluzione migliore consiste nel programmare l'esecuzione periodica mediante la funzionalità cron.

Per consentire l'esecuzione automatizzata della generazione della baseline, lo script genererà alcuni file di supporto, grazie ai quali potrà ripetere l'esecuzione senza prendere in considerazione periodi di tempo già scansionati.

Lo script è stato realizzato con Python 3.9 ed è testato sia in ambiente Windows che Linux; è compatibile con le versioni di QRadar 7.3 o superiori.

Alla prima esecuzione dello script viene richiesto all'utente l'inserimento di alcuni parametri come l'hostname o IP di QRadar, metodo di autenticazione, certificato e gruppi sorgente da tenere in considerazione; in seguito, se salvati, non sarà più necessario reinserirli.

Infine, considerato che all'interno dell'infrastruttura Cineca vi sono servizi di monitoraggio che periodicamente effettuano richieste nei confronti delle Web Application, al fine di evitare che possano essere rilevati come una minaccia dal sistema di rilevamento è stata stilata una WhiteList. L'implementazione di questa lista è stata gestita direttamente su QRadar configurando un apposito ReferenceSet che, associato a un BuildingBlock consente di non aprire Offense per gli IP presenti in WhiteList.

7.5 Limitazioni incontrate

Durante le fasi di sviluppo del progetto sono state incontrate alcune limitazioni che

sono state gestite come dettagliato di seguito.

L'idea iniziale era quella di creare uno strumento direttamente integrabile ed eseguibile su QRadar Console, tuttavia il CRE si basa su un set di criteri predefinito ed elabora gli eventi di volta in volta che arrivano; questo procedimento della pipeline non consente di generare Offense sulla base di analisi eseguite a posteriori. Inoltre, tramite le API RESTful, non è possibile generare una nuova Offense. Per questi motivi, al fine di consentire agli utenti di QRadar di ricevere una segnalazione nel pannello dedicato, si è scelto di inviare tramite syslog un formato di log personalizzato contenente i dettagli della minaccia. Grazie alla creazione di una regola specifica è poi possibile creare l'Offense alla ricezione di questa tipologia di log.

A livello di analisi dei comportamenti, i log che erano stati messi a disposizione non hanno consentito di svolgere i procedimenti in maniera agevole poiché hanno dato ampio spazio a casistiche particolari da tenere in considerazione. A titolo di esempio si può fare riferimento ai comportamenti che un attaccante può adottare riportati nella sezione 6.1.

Inoltre, è bene precisare che in alcune situazioni molto particolari, in cui l'utente malintenzionato conosca esattamente come viene svolta l'analisi e la rilevazione potrebbe adottare comportamenti in grado di deviare l'analisi lentamente al fine di far apprendere alla baseline anche i comportamenti anomali che, di conseguenza, verranno trattati come normali.

7.6 Sviluppi futuri

I possibili sviluppi futuri delle soluzioni proposte in questo progetto di tesi riguardano i modi in cui viene tracciata la generazione della sessione. Nello specifico il server web dovrebbe essere in grado di trasmettere, per ogni nuova sessione generata, l'id della sessione, il SourceIP a cui è stata assegnata e il VirtualHostName a cui è riferita. In questo modo si avrebbe assoluta certezza del numero di sessioni generate per conto di un determinato utente e servizio di

destinazione. Sulla base di tali informazioni sarà possibile semplificare notevolmente le procedure di analisi e renderle più precise. Inoltre, grazie a questa informazione, anche per gli utenti malintenzionati che conoscono come viene eseguita l'analisi sarà molto più difficile aggirare il controllo.

Sulla base delle indicazioni fornite si potrebbe ottenere un sistema più sicuro e performante.

8 Conclusione

Al giorno d'oggi la stragrande maggioranza delle applicazioni web fanno uso di sessioni applicative. Una caratteristica fondamentale del protocollo HTTP è quella di essere stateless; ossia un web server non ha alcun meccanismo per mantenere lo stato conversazionale dell'applicazione cioè per legare richieste provenienti da uno stesso client. Ogni richiesta che arriva a un web server è assolutamente indipendente dall'altra anche se proviene dal medesimo client. Questo non è un problema nel caso di siti statici anzi è un vantaggio in termini di prestazioni e di carico del server, ma lo è invece nel caso di applicazioni nelle quali quasi sempre si ha la necessità di legare logicamente richieste successive di uno stesso client. La gestione della sessione utente è quindi uno degli aspetti peculiari delle applicazioni web che le distingue dai siti web statici.

In questo progetto ci si è concentrati sul tracciamento delle sessioni utente al fine di evitare che i malintenzionati possano rendere indisponibile una Web Application esaurendo le sessioni disponibili.

Per quanto un server web possa essere potente e performante, avrà sempre dei limiti, specialmente al numero di richieste che può gestire in un determinato istante. Un'organizzazione che vuole fornire un servizio sempre disponibile alla sua utenza può utilizzare tecniche di Data Analysis per mitigare gli attacchi, una pratica che consente di analizzare i dati passati per prevedere situazioni future.

Gli attacchi basati sull'esaurimento delle sessioni applicative sono anche conosciuti come Session-Based Denial of Service e, siccome al momento non è possibile risolvere le limitazioni presenti, una soluzione che è possibile adottare in risposta al rilevamento di una potenziale minaccia è quella di inserire l'IP dell'attaccante in una blacklist in modo tale da mitigare le conseguenze.

9 Bibliografia e sitografia

[1] https://clusit.it/rapporto-clusit/

[2] https://www.cisco.com/c/it_it/products/security/common-cyberattacks.html

[3] https://www.ibm.com/it-it/topics/cyber-attack

[4] https://blog.t-consulting.it/i-10-cyber-attacchi-piu-comuni

[5] https://www.techopedia.com/definition/26306/error-log

[6] A. D. Calì, M. Patella, P. Ciaccia, Gestione delle Transazioni, Tecnologie delle Basi di Dati M - Alma Mater Studiorum Università di Bologna, Bologna 2014

[7] https://www.intelligenzaartificiale.it/sicurezza-informatica-e-intelligenza-artificiale/

[8] https://www.computer.org/publications/tech-news/trends/the-use-of-artificial-intelligence-in-cybersecurity

[9] S. Costantino, L. Deri “Rilevazione di anomalie di rete mediante analisi su serie temporali”, Università degli Studi di Pisa, Pisa, 2019

[10] https://securityboulevard.com/2021/07/what-is-anomaly-detection-in-cybersecurity/

[11] http://www.mokabyte.it/2007/02/sessiontracking/

[12] https://www.ibm.com/docs/en/qsip/7.3.3