Report finale M5- Splunk

Introduzione

In questo esame, dobbiamo andare ad analizzare dei dati tramite Splunk.

Splunk non è altro che un software progettato per raccogliere, indicizzare, cercare, analizzare e visualizzare grandi quantità di dati generati da macchine in tempo reale, tipicamente sotto forma di log.

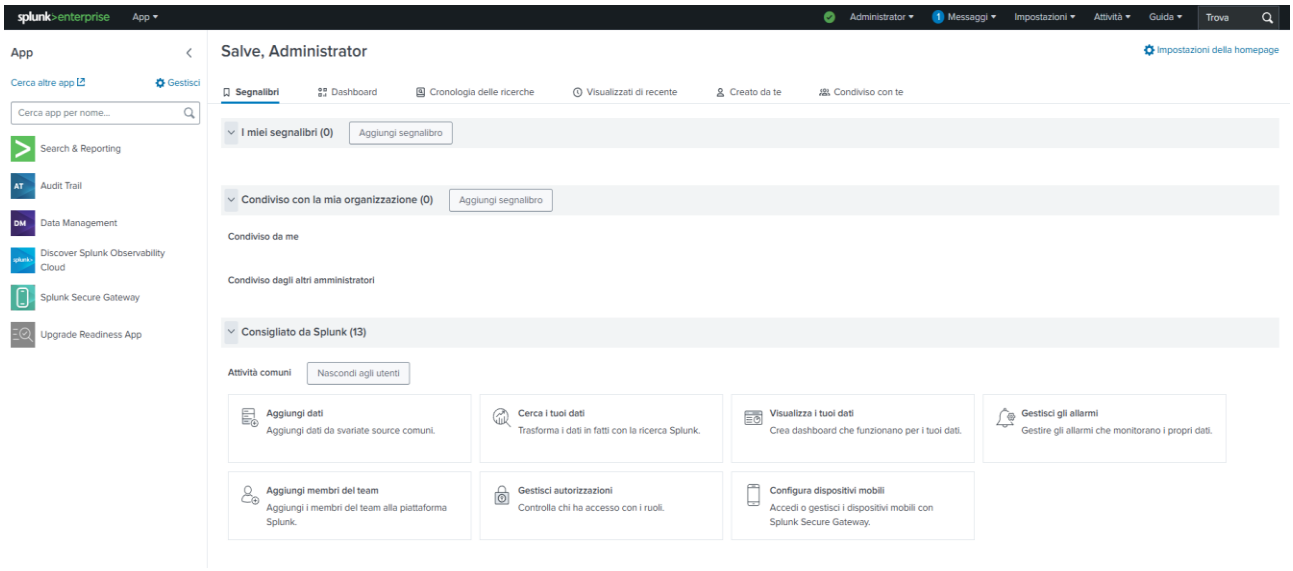
Splunk viene usato per monitorare eventi di sistema, rete e applicazioni, individuare anomalie e incidenti di sicurezza, creare dashboard/report interattivi ed automatizzare alert e investigazioni forensi.

Tecnicamente, Splunk prende dati non strutturati (log di server, firewall, router, applicazioni, ecc.), li indicizza e permette di interrogarli con un linguaggio di ricerca specifico chiamato SPL (Search Processing Language), rendendo più veloce e semplice trovare correlazioni o pattern sospetti.

Oltre a Splunk ci servirà la cartella .zip **tutorialdata**

Caricare i dati su Splunk

Una volta scaricato ed installato Splunk bisognerà avviarlo, trovandoci questa schermata



Schermata di Splunk

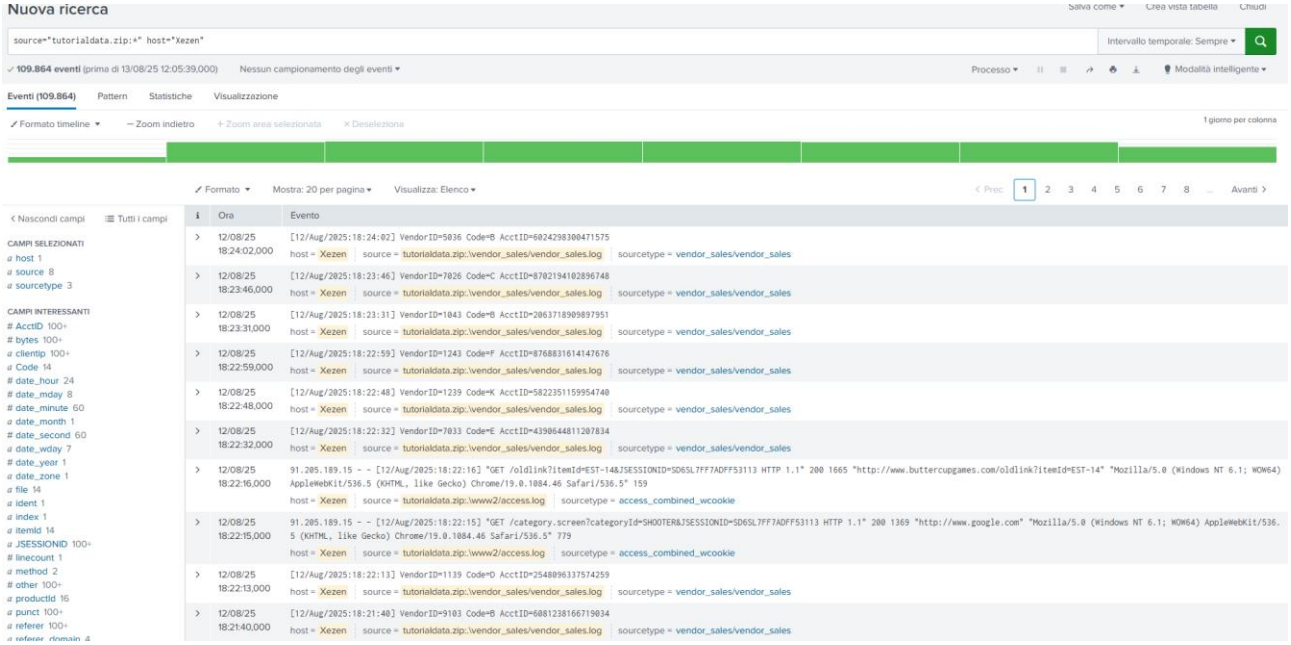
In basso a destra selezioniamo aggiungi dati

Upload del file

Selezioniamo la "Carica" ed andiamo a importarci la cartella **tutorialdata.zip** trascinandola direttamente o selezionandola.

Una volta selezionati i parametri avviamo la ricerca

Ci ritroviamo nella schermata principale di Splunk



L'ambiente Splunk mostrato in figura evidenzia l'avvenuta importazione e indicizzazione del dataset tutorialdata.zip, contenente log provenienti da più sorgenti, in particolare:

File di log delle vendite (vendor_sales/vendor_sales.log)
Log di accesso web (access_combined_wcookie)

Nel pannello "Search & Reporting" sono visibili **109.864 eventi** indicizzati, filtrati per la sorgente tutorialdata.zip sul sistema host XEZEN. La visualizzazione a elenco mostra, per ciascun evento, dettagli come:

Ora e data dell'evento
Host di origine
Percorso del file sorgente
Sorgente e tipo di log (sourcetype)
Campi specifici estratti automaticamente

Il grafico temporale in alto mostra la distribuzione degli eventi nel periodo selezionato (“Sempre”), permettendo di identificare picchi di attività. Questa configurazione è tipica di un dataset di laboratorio destinato all’esercizio di query SPL e analisi forense, in cui l’obiettivo è individuare pattern anomali, tentativi di accesso, o errori applicativi a partire dai log grezzi.

Punto 1: Crea una query Splunk per identificare tutti i tentativi di accesso falliti "Failed password". La query dovrebbe mostrare il timestamp, l'indirizzo IP di origine, il nome utente e il motivo del fallimento.

Come da titolo della richiesta dobbiamo andare ad analizzare i tentativi di accesso falliti con annessi Timestamp, IP d’origine, Username e la causa del fallimento. Andiamo a scrivere la nostra query:

```
source="tutorialdata.zip:*" "Failed password"
| rex "(?<failure_reason>Failed password for (?:invalid user )?(?<user>\S+)) from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3})"
| table _time src_ip user failure_reason
| sort _time
```

Spiegazione della query

(**Riga 2)**

source="tutorialdata.zip:*": è un campo di Splunk che indica il file o la posizione da cui proviene il log. In questo caso prendi tutti i file provenienti dal pacchetto tutorialdata.zip. L’asterisco * è un wildcard che include qualsiasi file all’interno dell’archivio;

“failed password”: è una stringa di testo che Splunk deve cercare all’interno del contenuto degli eventi;

| rex: il pipe serve per passare il risultato della ricerca al comando successivo mentre rex è il comando di Splunk che ti permette di estrarre campi da una stringa usando una regex.

"(?<failure_reason>: Crea un campo chiamato failure_reason che conterrà tutto ciò che è tra le parentesi.

Failed password for: parte fissa della frase presente nel log.

(?:invalid user)?: gruppo non catturante che, se presente, prende la frase "invalid user " (può esserci o no, per questo c’è il ?)

(?<user>\S+)): crea un campo chiamato user che prende la prossima sequenza di caratteri non spazi (\S+), cioè il nome utente.

from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3})": crea il campo src_ip che cattura un IPv4 sorgente (che nel log si trova dopo la parola “from”)

(riga 3)

TABLE: è un comando di formattazione in Splunk che serve per mostrare solo i campi che ti interessano e disporli in forma tabellare, escludendo tutto il resto.

_time: è campo di sistema di Splunk che contiene la data e ora dell’evento.

Src_ip: è il campo che è stato creato prima tramite la regex. Contiene l’indirizzo IP sorgente del tentativo di accesso.

User: è un Campo (estratto con la regex) che rappresenta il nome utente con cui si è tentato l’accesso.

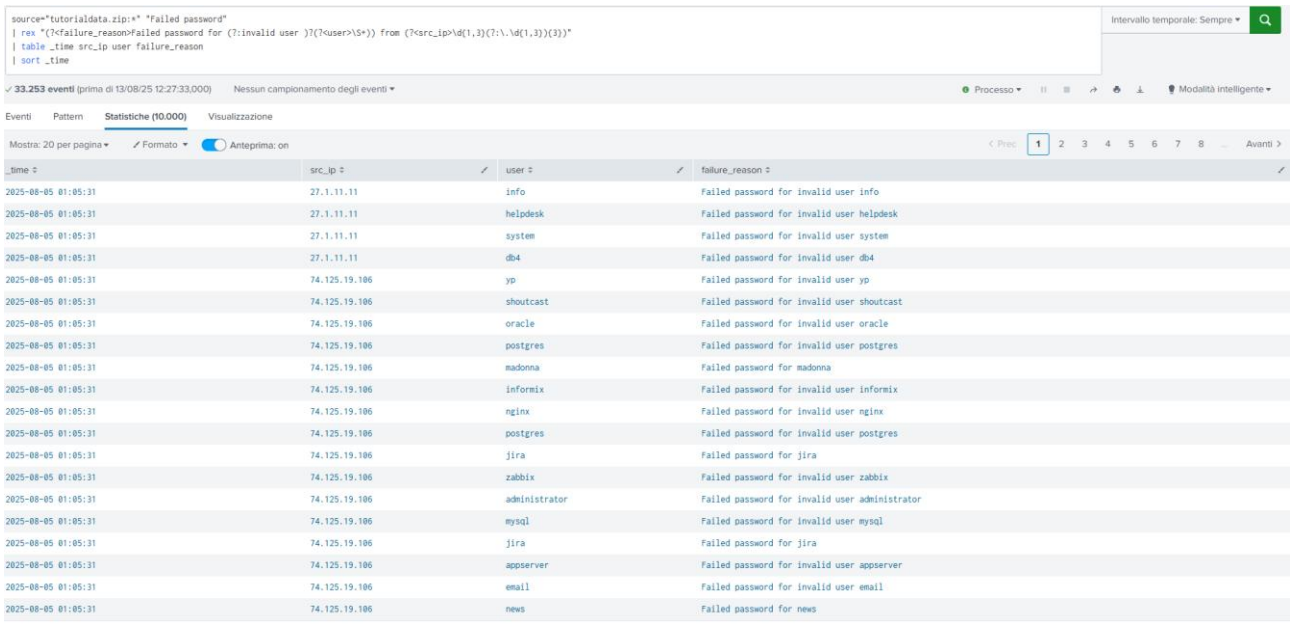
Failure_reason semplicemente contiene la frase del fallimento.

(Riga 3)

| Sort è il comando in Splunk che ordina i risultati in base a uno o più campi.

_time è un elemento di sistema che indica data e ora dell'evento.

Ci ritroveremo il seguente output:



```
source="tutorialdata.zip:*" "Failed password"
| rex "(?<failure_reason>Failed password for (?:invalid user )?(?<user>\S+)) from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3})"
| table _time src_ip user failure_reason
| sort _time
```

Intervallo temporale: Sempre

✓ 33.253 eventi (prima di 13/08/25 12:27:33,000) Nessun campionamento degli eventi

Processo Modalità intelligente

Eventi Pattern Statistiche (10.000) Visualizzazione

Mostra: 20 per pagina Formato Anteprima: on

< Prec 1 2 3 4 5 6 7 8 ... Avanti >

_time	src_ip	user	failure_reason
2025-08-05 01:05:31	27.1.11.11	info	Failed password for invalid user info
2025-08-05 01:05:31	27.1.11.11	helpdesk	Failed password for invalid user helpdesk
2025-08-05 01:05:31	27.1.11.11	system	Failed password for invalid user system
2025-08-05 01:05:31	27.1.11.11	db4	Failed password for invalid user db4
2025-08-05 01:05:31	74.125.19.106	yp	Failed password for invalid user yp
2025-08-05 01:05:31	74.125.19.106	shoutcast	Failed password for invalid user shoutcast
2025-08-05 01:05:31	74.125.19.106	oracle	Failed password for invalid user oracle
2025-08-05 01:05:31	74.125.19.106	postgres	Failed password for invalid user postgres
2025-08-05 01:05:31	74.125.19.106	madonna	Failed password for madonna
2025-08-05 01:05:31	74.125.19.106	informix	Failed password for invalid user informix
2025-08-05 01:05:31	74.125.19.106	nginx	Failed password for invalid user nginx
2025-08-05 01:05:31	74.125.19.106	postgres	Failed password for invalid user postgres
2025-08-05 01:05:31	74.125.19.106	jira	Failed password for jira
2025-08-05 01:05:31	74.125.19.106	zabbix	Failed password for invalid user zabbix
2025-08-05 01:05:31	74.125.19.106	administrator	Failed password for invalid user administrator
2025-08-05 01:05:31	74.125.19.106	mysql	Failed password for invalid user mysql
2025-08-05 01:05:31	74.125.19.106	jira	Failed password for jira
2025-08-05 01:05:31	74.125.19.106	appserver	Failed password for invalid user appserver
2025-08-05 01:05:31	74.125.19.106	email	Failed password for invalid user email
2025-08-05 01:05:31	74.125.19.106	news	Failed password for news

Azioni di rimedio

L'analisi dei log ha evidenziato un attacco massivo di tipo **brute force distribuito** mirato a compromissione di credenziali SSH. Per mitigare il rischio e rafforzare la postura di sicurezza del sistema, si raccomandano i seguenti interventi, suddivisi per area di applicazione.

1. Contromisure a livello di rete

Blocco automatico degli IP malevoli: implementare strumenti come *Fail2Ban* o moduli equivalenti in firewall/IDS, impostando regole che bannano automaticamente un IP dopo un numero predefinito di tentativi falliti (es. 5-10 tentativi in 1 minuto).

Rate limiting: configurare il firewall per limitare il numero di connessioni SSH possibili per IP in un intervallo di tempo, riducendo l'efficacia degli attacchi a dizionario.

Whitelist di accesso SSH: permettere l'accesso SSH solo da specifici indirizzi IP o subnet fidate (VPN aziendali, IP statici autorizzati).

Port knocking / cambio porta SSH: anche se non è una misura definitiva, modificare la porta SSH standard (22) o utilizzare il port knocking può ridurre l'esposizione agli scanner automatici.

2. Sicurezza degli account

Eliminazione account inutilizzati: rimuovere o disattivare utenze non più necessarie per ridurre la superficie di attacco.

Ridenominazione degli account amministrativi predefiniti: rinominare root, admin o administrator per ostacolare attacchi automatizzati che usano liste predefinite di nomi.

Policy di password robuste: imporre password lunghe (≥ 12 caratteri) con combinazione di maiuscole, minuscole, numeri e caratteri speciali, vietando parole comuni o pattern ricorrenti.

Cambio credenziali post-attacco: forzare un cambio password immediato su tutti gli account che hanno ricevuto tentativi di login, anche se non riusciti, per prevenire attacchi successivi.

3. Autenticazione avanzata

MFA (Multi-Factor Authentication): abilitare un secondo fattore di autenticazione (token hardware, app mobile, OTP via SMS) sugli account con privilegi elevati o accesso remoto.

Chiavi SSH al posto delle password: disabilitare l'autenticazione basata su password e permettere solo l'accesso tramite coppie di chiavi pubblica/privata con passphrase sicura.

4. Monitoraggio e risposta

Correlazione log: verificare se, successivamente ai tentativi falliti, ci siano stati login riusciti dagli stessi IP o intervalli temporali.

Alert in tempo reale: configurare Splunk o un SIEM per generare notifiche immediate in caso di attività anomala (ad esempio più di 5 tentativi falliti in un minuto da uno stesso IP).

Blacklist IP malevoli: confrontare gli IP sorgenti con database OSINT come *AbuseIPDB* o *AlienVault OTX* e bloccare quelli già segnalati.

Retention e conservazione forense: conservare i log completi per almeno 90 giorni per eventuali indagini retrospettive.

5. Formazione e consapevolezza

Sensibilizzazione del personale tecnico: formare gli amministratori sull'analisi dei log e sull'uso di strumenti di difesa proattiva.

Procedure di incident response: avere un piano di risposta agli incidenti aggiornato, con ruoli e responsabilità chiari in caso di compromissione.

Analisi e considerazione finale

L'analisi dei log provenienti dall'archivio tutorialdata.zip, filtrati in Splunk per la stringa "Failed password", ha evidenziato **33.253 eventi** di autenticazione non riuscita, tutti concentrati nello stesso istante temporale (2025-08-05 01:05:31). Questo pattern è indicativo di un attacco automatizzato massivo, probabilmente condotto tramite script distribuiti o una botnet.

Gli indirizzi IP sorgenti coinvolti sono numerosi e distribuiti geograficamente. Tra i più attivi figurano **27.1.11.11**, **74.125.19.106**, **188.143.232.202**, **94.230.166.185**, **90.205.111.169** e **95.163.78.227**, ognuno dei quali ha tentato connessioni verso molteplici account.

Gli utenti presi di mira includono:
Account amministrativi e di sistema: root, admin, administrator

Account di servizio e database: postgres, oracle, mysql, nagios, tomcat, jboss
Account generici o fittizi: helpdesk, info, games, sales, harrypotter, madonna

Una grande parte degli eventi mostra il messaggio invalid user, il che suggerisce l'utilizzo di liste di nomi utente generate da dizionari. Tuttavia, sono presenti anche tentativi verso utenti validi con password errata, indicando una conoscenza parziale della configurazione del sistema da parte degli aggressori.

Punto 2: Scrivi una query Splunk per trovare tutte le sessioni SSH aperte con successo. La query dovrebbe filtrare per l'utente "djohnson" e mostrare il timestamp e l'ID utente.

Arrivati a questo punto facciamo un check delle sessioni SSH aperte. Questo perché la porta SSH (Secure Shell) permette l'accesso non che controllo da remoto di un dispositivo.

Cominciamo con la scrittura della query:

```
index="main"
source="tutorialdata.zip:*"
| rex "pam_unix\((?<protocol>[^:]+):session\): (?<session_result>session [^ ]+) for user (?<username>\S+)"
| search username="djohnson" session_result="session opened*" protocol="sshd"
| table _time username uid protocol session_result
```

Analisi query:
(Riga 3)

| rex "pam_unix\(: passa i risultati al comando rex, che usa una regex per estrarre campi da _raw.

(?<protocol>[^:]+): salva nel campo protocol tutto fino ai due punti prima di **:session.**

(?<session_result>session [^]+): salva in session_result la coppia di parole che inizia con session.

(?<username>\S+): salva in username la parola immediatamente dopo for user (nessuno spazio → \S+), cioè l'utente.

(riga 4)

| search: applica un filtro sui campi appena estratti.

username="djohnson": prende solo le righe dell'utente djohnson.

session_result="session opened*": mantiene solo le sessioni aperte.

protocol="sshd": limita ai log di sshd.

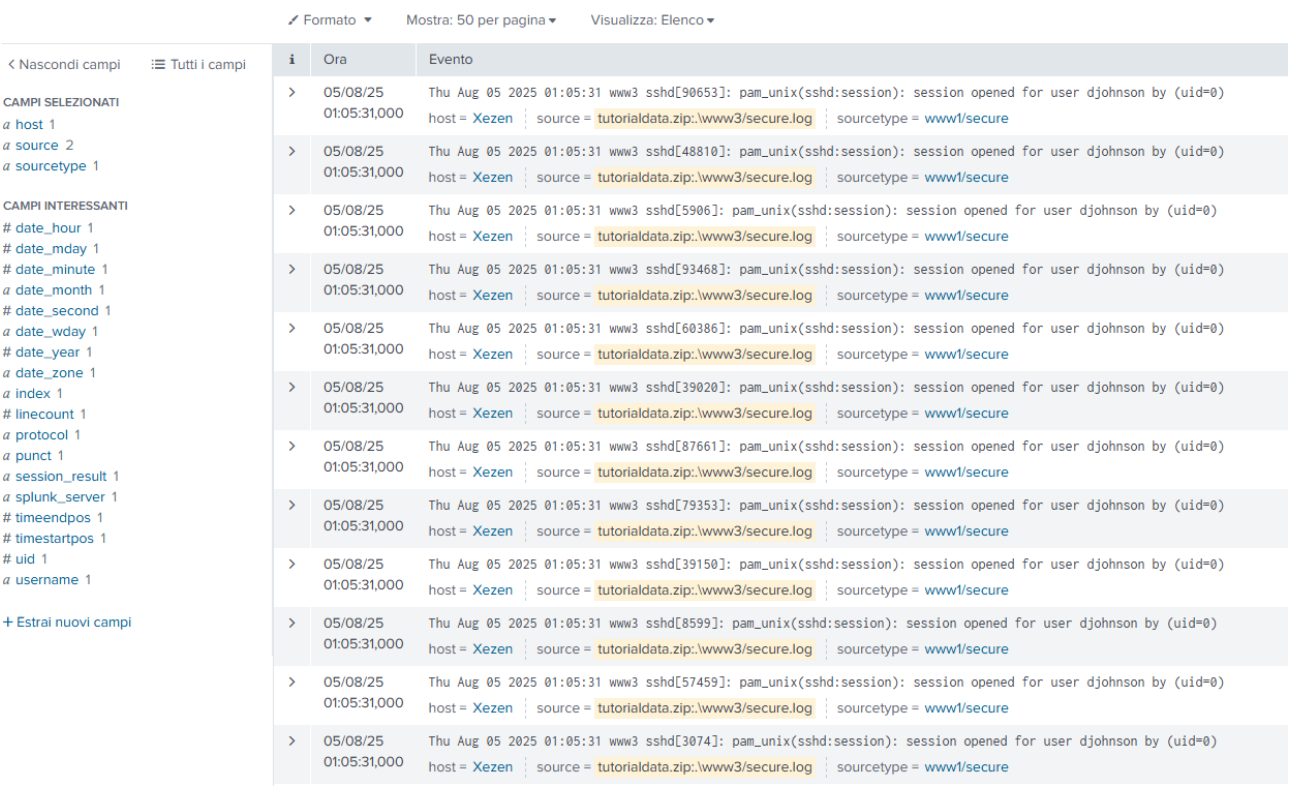
(riga 5)

| table: mostra i risultati in tabella con i soli campi indicati.

_time: stampa data ed ora dell'evento.

username, protocol, session_result: sono i dati che estrarrà.

Il risultato dato è il seguente:



Analisi e considerazione finale

Dall'analisi dei log riportati nello screenshot emerge una situazione in cui l'utente "djohnson" ha aperto più sessioni SSH nello stesso istante temporale, esattamente il 5 agosto 2025 alle ore 01:05:31. Ogni evento registrato proviene dal file di sicurezza **secure.log**, mostrando il messaggio generato dal modulo PAM **pam_unix** per il servizio sshd che conferma l'avvenuta apertura di sessione. Il dettaglio "**by (uid=0)"** rivela che l'autenticazione e l'avvio della sessione sono stati eseguiti da un account con privilegi di root, quindi con il massimo livello di autorizzazione sul sistema. La presenza di numerosi processi con identificativi differenti ma con lo stesso timestamp suggerisce un'attività anomala: in un contesto operativo normale, le aperture di sessione da parte dello stesso utente non avvengono simultaneamente e, quando accade, si tratta di un numero molto ridotto di connessioni parallele. Qui, invece, il volume e la simultaneità degli eventi fanno pensare a uno script automatizzato, a un processo malevolo o a un'azione di forza bruta che sfrutta credenziali già valide per moltiplicare l'accesso in più sessioni contemporanee.

Le conseguenze di un comportamento del genere possono essere significative. Un utente con credenziali legittime e privilegi elevati che apre numerose sessioni simultanee può eseguire comandi in parallelo, trasferire rapidamente dati sensibili, alterare configurazioni o installare malware senza che un'unica sessione sia facilmente individuabile come fonte dell'attacco. L'uso di UID 0, in particolare, aumenta esponenzialmente il rischio perché qualsiasi operazione eseguita ha un impatto diretto e totale sul sistema. In scenari reali, questo pattern di log potrebbe corrispondere a una compromissione in corso, con un attore malevolo che si assicura accessi multipli per rendere più difficile la sua rimozione o il tracciamento.

Per affrontare un'anomalia simile, l'azione immediata dovrebbe concentrarsi sull'identificazione dell'origine degli accessi, incrociando queste informazioni con i log di rete per risalire all'indirizzo IP di provenienza. Sarebbe fondamentale verificare se per ogni apertura di sessione corrisponde una chiusura nei log, così da individuare eventuali connessioni persistenti. Parallelamente, occorrerebbe esaminare la cronologia dei comandi eseguiti dall'utente in quel periodo e, se non strettamente necessario, sospendere o disabilitare

temporaneamente l'account "djohnson" per prevenire ulteriori accessi. L'aggiornamento delle credenziali e, se possibile, l'implementazione di meccanismi di autenticazione a più fattori costituirebbero un passo importante per rafforzare la sicurezza. Infine, la creazione di regole di alerting in Splunk permetterebbe di rilevare automaticamente aperture di sessione multiple nello stesso secondo, intervenendo in tempo reale in caso di nuove occorrenze.

In conclusione, ciò che emerge da questo frammento di log è un chiaro indicatore di attività sospetta che, in un contesto produttivo, non può essere considerata normale amministrazione di sistema. La combinazione di accessi simultanei, privilegi di root e ripetitività del pattern è un segnale di compromissione che richiede un'analisi approfondita e interventi immediati. La gestione tempestiva di episodi come questo è essenziale per limitare i danni, prevenire l'esfiltrazione di dati e garantire l'integrità dell'infrastruttura informatica.

Punto 3: Scrivi una query Splunk per trovare tutti i tentativi di accesso falliti provenienti dall'indirizzo IP "86.212.199.60". La query dovrebbe mostrare il timestamp, il nome utente e il numero di porta.

A questo punto, ho riscontrato il mio primo problema, nella versione del file che ho scaricato dal sito ufficiale, quindi file originale non ho l'indirizzo IP richiesto dalla prova. Per accertarmene ho fatto varie query di controllo senza alcun riscontro con l'IP 86.212.199.60 e per conferma finale ho creato una query che mi estrasse tutti gli IPv4 presenti all'interno del file confermando i miei dubbi come da immagine sottostante.

```
index=* source="tutorialdata.zip*"
| rex max_match=0 "(?<any_ip>\d{1,3}(?:\.\d{1,3}){3})"
| search any_ip="*.*.*.*"
| table _time source sourcetype any_ip _raw
| sort _time
```

2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	87.194.216.51	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www1/secure.log	www1/secure	90.205.111.169	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	91.205.189.27	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	91.205.189.27	Thu Aug 05 2025 01:05
2025-08-05 01:05:31	tutorialdata.zip:.\www3/secure.log	www1/secure	91.205.189.27	Thu Aug 05 2025 01:05

Quindi andrò a svolgere il punto su un altro IP scelto casualmente da me: **91.208.184.24**

```
index=main
source="tutorialdata.zip:*"
| rex "(?<failure_reason>Failed password for (?:invalid user )?(?<user>\S+)) from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3}) port (?<src_port>\d+)"
| search failure_reason="Failed password*" src_ip="91.208.184.24"
| table _time src_ip user src_port
```

Spiegazione Query:

(riga 3)

| rex: per estrarre i campi dal testo grezzo.

(?<failure_reason>Failed password for (?:invalid user)?(?<user>\S+)): crea **failure_reason** catturando l'intera frase **Failed password for invalid user admin** oppure **Failed password for root**.

(?:invalid user)?: è opzionale e gestisce sia utenti invalidi sia validi.

(?<user>\S+): crea il campo **user** con la parola subito dopo.

from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3}): estrae **src_ip** in formato IPv4.

port (?<src_port>\d+): estrae **src_port** (numerica). Qui la porta è obbligatoria: se in qualche riga non c'è "port …", quell'evento non estrarrà **src_port.**

(riga 4)

| search: applica un filtro sui campi appena estratti.

failure_reason="Failed password*": confronta il prefisso **failed password.**

src_ip="91.208.184.24": fa si che tiene solo gli eventi con quell'IP sorgente.

(riga5)

| table _time src_ip user src_port: mostra in tabella solo gli **user** (utenti)**, port** (porte)**, time** (ora e data), **src_ip** (IPv4).

_time	src_ip	user	src_port
2025-08-05 01:05:31	91.208.184.24	jira	1940
2025-08-05 01:05:31	91.208.184.24	henri	4801
2025-08-05 01:05:31	91.208.184.24	jboss	4281
2025-08-05 01:05:31	91.208.184.24	root	3283
2025-08-05 01:05:31	91.208.184.24	apache	4698
2025-08-05 01:05:31	91.208.184.24	helpdesk	1227
2025-08-05 01:05:31	91.208.184.24	sync	3445
2025-08-05 01:05:31	91.208.184.24	squid	3983
2025-08-05 01:05:31	91.208.184.24	jira	4196
2025-08-05 01:05:31	91.208.184.24	email	1735
2025-08-05 01:05:31	91.208.184.24	sys	1263
2025-08-05 01:05:31	91.208.184.24	mail	1121
2025-08-05 01:05:31	91.208.184.24	services	3371
2025-08-05 01:05:31	91.208.184.24	info	4371
2025-08-05 01:05:31	91.208.184.24	root	3526
2025-08-05 01:05:31	91.208.184.24	vmware	2123
2025-08-05 01:05:31	91.208.184.24	postgres	1140
2025-08-05 01:05:31	91.208.184.24	trac	1558
2025-08-05 01:05:31	91.208.184.24	apache	1304
2025-08-05 01:05:31	91.208.184.24	root	1594
2025-08-05 01:05:31	91.208.184.24	services	1501
2025-08-05 01:05:31	91.208.184.24	daemon	2330
2025-08-05 01:05:31	91.208.184.24	root	2381
2025-08-05 01:05:31	91.208.184.24	abc	3039
2025-08-05 01:05:31	91.208.184.24	email	1405
2025-08-05 01:05:31	91.208.184.24	mail	1738

Analisi e considerazione finale

Questa query ci permette dunque di avere tutti i dati che ci servono con target solo l'IP 91.208.184.24 con seguente output:Il report generato dalla query mostra un elenco di 26 tentativi di autenticazione SSH falliti, tutti concentrati nello stesso istante temporale, il 5 agosto 2025 alle 01:05:31, e provenienti dallo stesso indirizzo IP sorgente, **91.208.184.24**. Gli username tentati includono sia account di sistema ad alta criticità, come *root*, *postgres*, *vmware* e *daemon*, sia account di servizio come *apache*, *mail*, *helpdesk*, *services* o *jira*. Questa varietà indica un attacco a forza bruta mirato a sfruttare credenziali deboli o di default, nella speranza di ottenere accesso privilegiato o a servizi sensibili. La simultaneità di tutti gli eventi suggerisce che il traffico non sia stato generato manualmente, ma piuttosto da uno script o da un tool automatizzato, probabilmente eseguito da un sistema compromesso o parte di una botnet.

Dal punto di vista delle conseguenze, un attacco di questo tipo, se andasse a buon fine anche su uno solo degli account target, potrebbe permettere all'attaccante di ottenere accesso remoto non autorizzato, con possibilità di esfiltrare dati, installare malware persistente o sfruttare la macchina come pivot per muoversi lateralmente nella rete interna. Il fatto che siano presenti tentativi su account come *root* o *postgres* evidenzia un interesse verso privilegi elevati, mentre account applicativi come *jira* o *apache* potrebbero essere usati come punto di ingresso per ulteriori escalation.

Le azioni consigliate includono il blocco immediato dell'indirizzo IP 91.208.184.24 a livello di firewall o sistema di prevenzione intrusioni, la revisione dei log per individuare eventuali tentativi riusciti correlati, l'abilitazione dell'autenticazione a più fattori sugli account esposti e l'adozione di meccanismi di *rate limiting* o *fail2ban* per limitare i tentativi ripetuti. È inoltre fondamentale verificare che tutti gli account abbiano password robuste e non di default, e valutare la disabilitazione dell'accesso SSH diretto per gli utenti amministrativi.

In conclusione, il pattern osservato è indicativo di un'attività malevola sistematica, mirata a colpire diversi punti di accesso con un'unica ondata di richieste automatizzate. La rapidità nell'individuazione e nella risposta a questi eventi è cruciale per ridurre la superficie di attacco e prevenire compromissioni più gravi. Un'analisi retrospettiva su log storici e la messa in atto di regole di rilevamento proattive consentiranno di migliorare la postura di sicurezza e ridurre il rischio di attacchi simili in futuro.

Punto 4: Crea una query Splunk per identificare gli indirizzi IP che hanno tentato di accedere ("Failed password") al sistema più di 5 volte. La query dovrebbe mostrare l'indirizzo IP e il numero di tentativi.

Arrivati a questo punto, dando ormai per certo che si tratti di un attacco bruteforce/dizionario sulle porte vediamo tutti gli indirizzi IP che hanno fallito il login minimo 5 volte.

La query che ho utilizzato è la seguente:

```
index=main source="tutorialdata.zip:*" "Failed password"
| rex "from (?<src_ip>\d{1,3}(?:\.\d{1,3}){3})"
| stats count AS tentativi by src_ip
| where tentativi >= 5
| sort - tentativi
```

Spiegazione Query:

(riga 2)

| rex "from (?<src_ip>: cerca la parola **from** seguita da un IPv4 e salva l'IP nel campo **src_ip.**

\d{1,3}(?:\.\d{1,3}){3}: è un pattern "pragmatico" per IPv4 (accetta 0–999 per ottetto).

(riga 3)

| stats: raggruppa tutti gli eventi per **src_ip** e li conta**.**

count AS tentativi: crea una metrica numerica chiamata **tentativi** con il numero di eventi "**Failed password**" osservati per ciascun IP.

by src_ip: è la chiave di aggregazione (una riga per ogni IP).

(riga 4)

| where filtra le righe prodotte da **| stats.**

tentativi >= 5: mantiene solo gli IP che hanno almeno 5 tentativi di autenticazione falliti e la condizione è valutata numericamente (non come stringa), quindi è un vero confronto numerico.

(riga 5)

| sort – tentativi: sort ordina il risultato in base al campo tentativi con - davanti al nome del campo che indica l'ordine decrescente (gli IP più "rumorosi" in cima).

In pratica la query prende tutti i messaggi nei log in cui ci sono stati tentativi di accesso falliti, guarda da quali indirizzi IP sono arrivati e conta quante volte ogni IP ha provato ad accedere senza successo.

Poi elimina dalla lista tutti quelli che hanno fatto meno di cinque tentativi e mette in cima quelli che hanno provato più spesso. In pratica serve per individuare subito gli indirizzi IP più sospetti e potenzialmente pericolosi perché insistono ripetutamente nel tentare di entrare nel sistema.

src_ip	tentativi
128.241.220.82	40
87.194.216.51	37
194.215.205.19	36
216.221.226.11	30
91.208.184.24	26
211.191.168.25	25
118.142.68.222	24
188.138.40.166	24
211.166.11.101	21
76.89.103.115	21
110.159.208.78	20
201.42.223.29	20
27.101.11.11	20
82.245.228.36	20
198.35.2.120	19
67.133.102.54	18
71.192.86.205	18
94.230.166.185	18
208.240.243.170	17
69.80.0.18	17
74.125.19.106	17
91.205.189.27	17
59.99.230.91	16
81.18.148.190	16
222.169.224.226	15
85.62.218.82	15
200.6.134.23	13
200.6.134.23	13
203.223.0.20	13
27.35.11.11	13
60.220.218.88	12
188.143.232.202	12
69.72.161.186	11
125.89.78.6	11
210.192.123.204	11
90.205.111.169	10
174.123.217.162	10
175.44.3.30	10
195.80.144.22	10
198.35.1.10	10
217.132.169.69	10
76.169.7.252	9
195.216.243.24	8
203.45.206.135	8
95.163.78.227	8
99.61.68.230	7
66.69.195.226	7
84.34.159.23	7
91.217.178.210	6
209.160.24.63	5
202.91.242.117	5
210.76.124.106	

Possiamo notare in output gli IP che hanno tentato l'accesso a sinistra ed a destra il numero di tentativi effettuati (con un minimo di 5 tentativi) in ordine di decrescente di failed password.

Analisi e considerazione finale

Il report evidenzia un numero significativo di indirizzi IP che hanno effettuato ripetuti tentativi di accesso falliti al sistema, con volumi che vanno da un minimo di cinque a un massimo di quaranta tentativi. La concentrazione di così tanti errori di autenticazione in un intervallo temporale ristretto è un indicatore chiaro di attività automatizzata, come attacchi a forza bruta o scansioni massive di credenziali, condotti con l'obiettivo di individuare account vulnerabili. Alcuni IP si distinguono per un numero particolarmente alto di tentativi, il che li rende candidati prioritari per un'azione di contenimento immediata.

La conseguenza diretta di questa attività, se non contrastata, è la possibilità che un aggressore riesca a indovinare o sfruttare credenziali deboli, ottenendo così accesso non autorizzato alla macchina o ai servizi esposti. Questo comporterebbe rischi elevati, come l'esfiltrazione di dati, l'installazione di malware persistente o l'uso del sistema compromesso come punto di partenza per ulteriori attacchi interni. Inoltre, la varietà e la provenienza geografica potenzialmente distribuita di questi IP fanno pensare a una campagna di attacchi condotta tramite una botnet, rendendo più complesso il tracciamento e l'attribuzione.

Per mitigare il rischio, è raccomandato bloccare tempestivamente gli IP più attivi tramite firewall o sistemi di prevenzione intrusioni, implementare meccanismi di blocco automatico dopo un certo numero di tentativi falliti (ad esempio con strumenti come fail2ban), e verificare che tutte le credenziali siano robuste, aggiornate e, ove possibile, protette da autenticazione a più fattori. È anche importante correlare questi eventi con altri log per verificare se, dopo i tentativi falliti, siano avvenuti accessi riusciti, il che potrebbe indicare una compromissione in corso.

In conclusione, i dati mostrano un pattern tipico di attacco distribuito e persistente. La velocità nell'individuazione e nella risposta è fondamentale per contenere i danni. Implementare politiche di sicurezza proattive e sistemi di monitoraggio continuo ridurrà notevolmente la probabilità che un attacco simile possa avere successo in futuro.

Punto 5: Crea una query Splunk per trovare tutti gli Internal Server Error

Arrivati a questo controlliamo anche se ci sono problemi interni al server, non per forza dovuti ad un attacco o un fattore esterno.

La query utilizzata da me è la seguente:

```
index=main source="tutorialdata.zip:*" status=500
| table _time host source status uri_path
| sort _time
```

(riga 2)

_time: timestamp dell'evento.

host: il server/app che ha generato il log.

source: il file sorgente dentro tutorialdata.zip

status: il codice HTTP (qui 500).

uri_path: il percorso richiesto (se il sourcetype lo estrae; in alcuni set può chiamarsi uri, request, o essere assente).

(riga 3)

| sort _time: ordina i risultati in ordine cronologico crescente.

Questo ci da in output tutti i log che hanno avuto un errore 500, ovvero un errore interno del server che impedisce il completamento della richiesta del client. Si tratta di un messaggio generico che segnala un malfunzionamento lato server, spesso causato da bug applicativi, configurazioni errate, servizi interni non disponibili o eccezioni non gestite. Pur non fornendo dettagli specifici al client, questo errore richiede un'analisi lato server per individuarne la causa. In ambito sicurezza, la ricorrenza di errori 500 può anche essere indice di tentativi di sfruttare vulnerabilità o di inviare input malevoli per compromettere il sistema.

Di fatti vediamo il seguente report:

Eventi Pattern **Statistiche (733)** Visualizzazione

Mostra: 100 per pagina ✓ Formato Anteprima: on

< Prec 1 2 3 4 5 6

_time	host	source	status	uri_path
2025-08-05 18:34:11	Xezen	tutorialdata.zip:.\www3/access.log	500	/cart.do
2025-08-05 18:34:12	Xezen	tutorialdata.zip:.\www3/access.log	500	/cart.do
2025-08-05 18:34:14	Xezen	tutorialdata.zip:.\www3/access.log	500	/oldlink
2025-08-05 18:42:26	Xezen	tutorialdata.zip:.\www3/access.log	500	/oldlink
2025-08-05 18:53:54	Xezen	tutorialdata.zip:.\www3/access.log	500	/product.screen
2025-08-05 19:07:21	Xezen	tutorialdata.zip:.\www3/access.log	500	/product.screen
2025-08-05 19:18:01	Xezen	tutorialdata.zip:.\www3/access.log	500	/product.screen
2025-08-05 19:24:51	Xezen	tutorialdata.zip:.\www2/access.log	500	/cart.do
2025-08-05 19:29:19	Xezen	tutorialdata.zip:.\www1/access.log	500	/product.screen
2025-08-05 19:35:02	Xezen	tutorialdata.zip:.\www2/access.log	500	/oldlink
2025-08-05 19:45:53	Xezen	tutorialdata.zip:.\www1/access.log	500	/category.screen
2025-08-05 19:56:13	Xezen	tutorialdata.zip:.\www2/access.log	500	/cart.do
2025-08-05 20:05:47	Xezen	tutorialdata.zip:.\www1/access.log	500	/oldlink
2025-08-05 20:30:34	Xezen	tutorialdata.zip:.\www3/access.log	500	/product.screen
2025-08-05 20:38:57	Xezen	tutorialdata.zip:.\www2/access.log	500	/product.screen
2025-08-05 20:42:04	Xezen	tutorialdata.zip:.\www2/access.log	500	/category.screen
2025-08-05 21:20:37	Xezen	tutorialdata.zip:.\www2/access.log	500	/oldlink
2025-08-05 21:40:04	Xezen	tutorialdata.zip:.\www1/access.log	500	/cart.do
2025-08-05 21:47:31	Xezen	tutorialdata.zip:.\www2/access.log	500	/oldlink
2025-08-05 21:47:34	Xezen	tutorialdata.zip:.\www2/access.log	500	/cart.do
2025-08-05 21:52:14	Xezen	tutorialdata.zip:.\www3/access.log	500	/oldlink
2025-08-05 22:26:28	Xezen	tutorialdata.zip:.\www3/access.log	500	/product.screen
2025-08-05 22:31:26	Xezen	tutorialdata.zip:.\www1/access.log	500	/oldlink
2025-08-05 23:10:56	Xezen	tutorialdata.zip:.\www2/access.log	500	/category.screen
2025-08-05 23:17:45	Xezen	tutorialdata.zip:.\www1/access.log	500	/oldlink
2025-08-05 23:21:05	Xezen	tutorialdata.zip:.\www2/access.log	500	/category.screen
2025-08-05 23:28:35	Xezen	tutorialdata.zip:.\www2/access.log	500	/oldlink
2025-08-05 23:32:21	Xezen	tutorialdata.zip:.\www2/access.log	500	/oldlink

Analisi e considerazione finale

L'analisi ha evidenziato un numero elevato di errori 500 distribuiti su più host e sorgenti di log, con particolare ricorrenza su endpoint come **/cart.do, /oldlink, /product.screen e /category.screen**. La persistenza degli errori su più giorni e in orari diversi suggerisce un problema strutturale e ricorrente, piuttosto che un evento isolato. Alcuni percorsi, come **/oldlink**, potrebbero indicare richieste verso risorse obsolete o non più supportate, mentre la ripetitività degli stessi errori su specifiche funzioni (es. carrello e pagine prodotto) fa pensare a malfunzionamenti applicativi o configurazioni errate dei servizi.

Le conseguenze di un Internal Server Error non vanno sottovalutate: oltre a compromettere la disponibilità del servizio e l'esperienza utente, un'alta frequenza di errori 500 può rappresentare un potenziale punto debole sfruttabile da un attaccante. Un input malevolo può provocare eccezioni non gestite che espongono dettagli interni del sistema o creano condizioni favorevoli per attacchi più complessi, come denial of service o tentativi di exploit mirati.

Per mitigare i rischi, si raccomanda un'analisi approfondita dei log di errore dell'applicazione per individuare eventuali eccezioni ricorrenti, la revisione delle configurazioni dei servizi web e dei moduli collegati, e la rimozione o l'aggiornamento dei link obsoleti. È inoltre opportuno implementare controlli di validazione e

gestione robusta degli errori lato server, e predisporre un monitoraggio continuo con alert automatici in Splunk per rilevare rapidamente picchi anomali.

In conclusione, la rilevazione e l'analisi sistematica degli Internal Server Error non solo migliora l'affidabilità e la stabilità del sistema, ma rappresenta anche un'azione preventiva fondamentale in ottica di sicurezza, riducendo la probabilità di compromissioni e garantendo tempi di risposta più rapidi in caso di incidenti.

Conclusioni finali

L'analisi complessiva condotta con Splunk sul dataset fornito ha permesso di individuare, correlare e interpretare diversi scenari di interesse sia in ottica di sicurezza informatica sia di stabilità dei servizi. L'approccio metodico, basato sulla scrittura di query mirate e sull'estrazione di campi significativi tramite regex, ha consentito di trasformare log grezzi in informazioni operative concrete, utili per l'individuazione tempestiva di anomalie e potenziali minacce.

Le evidenze raccolte mostrano un contesto caratterizzato da tentativi di accesso non autorizzati, pattern tipici di attacchi a forza bruta, utilizzo simultaneo e sospetto di sessioni SSH privilegiate, e la presenza ricorrente di errori interni al server (HTTP 500). Questi elementi, presi singolarmente, possono già rappresentare un rischio significativo; analizzati in correlazione, delineano uno scenario in cui le superfici di attacco sono multiple e richiedono un'attività costante di monitoraggio e risposta.

Dal punto di vista operativo, l'utilizzo di Splunk ha dimostrato l'efficacia di uno strumento SIEM nella gestione di grandi volumi di dati eterogenei, permettendo di isolare rapidamente gli eventi più critici, verificarne il contesto e definire strategie di mitigazione mirate. Il valore aggiunto di questa attività non risiede unicamente nella capacità di "fotografare" lo stato attuale, ma anche nel gettare le basi per un sistema di rilevamento proattivo, in grado di generare alert in tempo reale e di supportare processi di incident response ben strutturati.

In sintesi, il progetto conferma che un'analisi sistematica dei log, unita a competenze nella formulazione di query efficaci e alla conoscenza dei possibili scenari di minaccia, rappresenta un elemento imprescindibile per garantire sicurezza, affidabilità e resilienza dei sistemi. La lezione principale che si può trarre è che la velocità e la precisione nell'individuare i segnali di compromissione sono tanto importanti quanto la capacità di prevenire, attraverso configurazioni sicure, buone pratiche di gestione degli account e monitoraggio continuo.