Introduzione

In questo esercizio abbiamo importato su Splunk i dati di esempio “tutorialdata.zip”, contenenti log di autenticazione e traffico web, per analizzare accessi e errori di sistema. Successivamente, abbiamo creato delle query per delle situazioni specifiche e abbiamo analizzato il contenuto, i possibili attacchi e come evitarli.

Splunk

E’ una piattaforma per l’analisi di dati machine-generated in tempo reale. E’ ampiamente utilizzata per monitoraggio, sicurezza informatica, analisi operativa e molto altro.

Nel nostro caso lo usiamo per analizzare il file tutorialdata.zip che è il file ufficiale per imparare di splunk.

Domande e risultati.

● 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.

```
sourcetype="www1/secure" host="Javier" "Failed password" "86.212.199.60"
| rex field=_raw "for (?:invalid user )?(?<user>\S+)"
| rex field=_raw "port (?<port>\d+)"
| table _time, user, port
```

Per analizzare il codice andremo a dividerlo in parti.

sourcetype="www1/secure" host="Javier" "Failed password" "86.212.199.60"

Filtra i log per quelli che hanno il sourcetype uguale a "www1/secure" cioè il tipo di log,
host uguale a "Javier" la macchina da cui provengono
che contengono la stringa "Failed password" (tentativi di accesso falliti),
e che contengono anche l’indirizzo IP "86.212.199.60".

| rex field=_raw "for (?:invalid user)?(?<user>\S+)"

Usa il comando rex per estrarre dal campo _raw il nome utente.
La regex cerca la parola for seguita opzionalmente da invalid user e poi cattura il nome utente in un campo chiamato user.
\S+ significa “una sequenza di caratteri senza spazi”.

| rex field=_raw "port (?<port>\d+)"

Usa un’altra regex per estrarre il numero di porta dal testo _raw.
Cerca la parola port seguita da uno o più numeri (\d+) e li salva nel campo port

| table _time, user, port

Mostra una tabella con solo tre colonne:

_time → il timestamp dell’evento (quando è successo),
user → il nome utente estratto,
port → il numero di porta estratto.

Analizzando i log troviamo che molto probabilmente ha subito un attacco di brute force a dictionary alcuni dei segnali tipici sono:

- Numerosi tentativi in pochi minuti
- Username a cas
- Stesso indirizzo IP

Per ridurre il rischio di brute force sugli accessi si potrebbe:

- Limitare i tentativi di login
- Usare autenticazioni forti
- Filtrare gli IP sospetti
- Aggiornare regolarmente software e sistemi

	Ora	Evento
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5630]: Failed password for root from 86.212.199.60 port 1783 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5404]: Failed password for root from 86.212.199.60 port 4380 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3063]: Failed password for root from 86.212.199.60 port 4032 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3176]: Failed password for root from 86.212.199.60 port 2475 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5630]: Failed password for root from 86.212.199.60 port 1783 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5630]: Failed password for root from 86.212.199.60 port 1783 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5404]: Failed password for root from 86.212.199.60 port 4380 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5404]: Failed password for root from 86.212.199.60 port 4380 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3063]: Failed password for root from 86.212.199.60 port 4032 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure

_time	user	port
2025-08-07 10:50:38	root	1783
2025-08-07 10:50:38	email	4692
2025-08-07 10:50:38	sys	3263
2025-08-07 10:50:38	oracle	2004
2025-08-07 10:50:38	ventrilo	4391
2025-08-07 10:50:38	system	1436
2025-08-07 10:50:38	nagios	1309
2025-08-07 10:50:38	root	4380
2025-08-07 10:50:38	peevish	4938
2025-08-07 10:50:38	root	4032
2025-08-07 10:50:38	root	2475
2025-08-07 10:50:38	operator	4720
2025-08-07 10:50:38	inet	3588

● 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.

source="tutorialdata.zip:*" host="Javier" "Failed password" | rex field=_raw "Failed password for (?:invalid user)?(?<username>\S+) from (?<src_ip>\S+)"
| eval Failure_Reason="Failed password"
| table _time, src_ip, username, Failure_Reason

Analizziamo il codice.

source="tutorialdata.zip:*" host="Javier" "Failed password"

Cerca nei dati provenienti dal file tutorialdata.zip (tutti i file al suo interno *)
Solo gli eventi con host = Javier
Che contengono la stringa "Failed password".

| rex field=_raw "Failed password for (?:invalid user)?(?<username>\S+) from (?<src_ip>\S+)"

Usa una regex per estrarre due campi dal testo grezzo (_raw):
username → prende la parola subito dopo "Failed password for"
(?:invalid user)? significa "può esserci la frase 'invalid user ' o no".
\S+ cattura una stringa senza spazio.
src_ip → prende l'IP subito dopo la parola "from".

| eval Failure_Reason="Failed password"

Crea un nuovo campo chiamato Failure_Reason e gli assegna sempre il valore "Failed password".
Serve per avere il motivo in una colonna dedicata.

| table _time, src_ip, username, Failure_Reason

Mostra solo queste colonne in tabella:

_time → timestamp dell'evento.
src_ip → IP di origine.
username → nome utente.
Failure_Reason → motivo del fallimento.

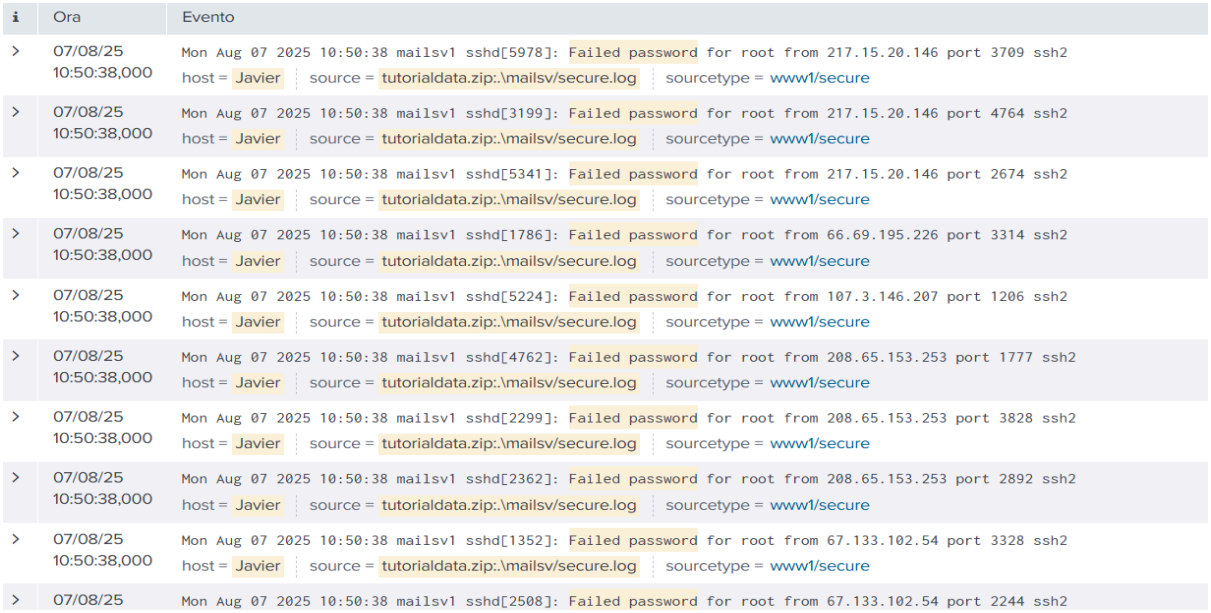
Anche qua analizzando i dati potrebbe trattarsi di un caso di brute force oppure credential stuffing alcuni dei segni tipici sono:

Molti tentativi in breve tempo.
Target sempre l'utente root.
IP diversi → evita i ban basati sull'IP singolo.
Tentativi su SSH → servizio spesso bersagliato.
Target root sul server mailsv1

Alcune best practice sono:

Disabilitare il login diretto di root via SSH
Usare solo chiavi
Limitare IP autorizzati
limitare i tentativi di login

_time	src_ip	username	Failure_Reason
2025-08-07 10:50:38	201.28.109.162	irc	Failed password
2025-08-07 10:50:38	201.28.109.162	mail	Failed password
2025-08-07 10:50:38	60.220.218.88	mailman	Failed password
2025-08-07 10:50:38	217.15.20.146	admin	Failed password
2025-08-07 10:50:38	217.15.20.146	root	Failed password
2025-08-07 10:50:38	217.15.20.146	tavi	Failed password
2025-08-07 10:50:38	217.15.20.146	root	Failed password
2025-08-07 10:50:38	217.15.20.146	info	Failed password
2025-08-07 10:50:38	217.15.20.146	sys	Failed password
2025-08-07 10:50:38	217.15.20.146	mysql	Failed password
2025-08-07 10:50:38	217.15.20.146	root	Failed password
2025-08-07 10:50:38	217.15.20.146	apache	Failed password
2025-08-07 10:50:38	217.15.20.146	mysql	Failed password
2025-08-07 10:50:38	217.15.20.146	whois	Failed password
2025-08-07 10:50:38	217.15.20.146	demon	Failed password
2025-08-07 10:50:38	66.69.195.226	peevish	Failed password



i	Ora	Evento
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5978]: Failed password for root from 217.15.20.146 port 3709 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3199]: Failed password for root from 217.15.20.146 port 4764 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5341]: Failed password for root from 217.15.20.146 port 2674 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[1786]: Failed password for root from 66.69.195.226 port 3314 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5224]: Failed password for root from 107.3.146.207 port 1206 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[4762]: Failed password for root from 208.65.153.253 port 1777 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[2299]: Failed password for root from 208.65.153.253 port 3828 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[2362]: Failed password for root from 208.65.153.253 port 2892 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[1352]: Failed password for root from 67.133.102.54 port 3328 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25	Mon Aug 07 2025 10:50:38 mailsv1 sshd[2508]: Failed password for root from 67.133.102.54 port 2244 ssh2

● 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.

source="tutorialdata.zip:*" "Failed password"
| rex "from (?<src_ip>\d+\.\d+\.\d+\.\d+)"
| stats count AS tentativi_falliti BY src_ip
| where tentativi_falliti > 5
| sort - tentativi_falliti
| table src_ip, tentativi_falliti

Analizziamo anche questo codice.

source="tutorialdata.zip:*" "Failed password"

Cerca nei file contenuti dentro tutorialdata.zip tutte le linee che contengono la stringa "Failed password" (tipico messaggio dei log SSH quando una password è sbagliata).

| rex "from (?<src_ip>\d+\.\d+\.\d+\.\d+)"

Usa una regex (rex) per estrarre l'indirizzo IP dal testo del log, prendendo ciò che appare dopo la parola "from ".

Salva il valore in un campo chiamato src_ip

| stats count AS tentativi_falliti BY src_ip

Conta il numero di volte (count) in cui ogni IP (src_ip) compare, e rinomina il conteggio come tentativi_falliti.

| where tentativi_falliti > 5

Filtra i risultati mostrando **solo gli IP** che hanno più di 5 tentativi falliti.

| sort - tentativi_falliti

Ordina i risultati in ordine decrescente di tentativi falliti.

| table src_ip, tentativi_falliti

Mostra solo due colonne nella tabella finale: src_ip e tentativi_falliti, per rendere la visualizzazione più chiara.

Anche qua potrebbe trattarsi di brute force a dictionary dovuti all'alta quantità di tentativi falliti e che cercano di collegarsi al servizio ssh.

In questo caso possono applicarsi i consigli di prima per evitare gli attacchi brute force

src_ip	tentativi_falliti
87.194.216.51	225
128.241.220.82	153
65.19.167.94	126
109.169.32.135	111
211.166.11.101	108
66.69.195.226	99
86.212.199.60	99
142.233.200.21	90
170.192.178.10	87
217.132.169.69	84
59.99.230.91	81
107.3.146.207	78
61.164.73.20	78
78.111.167.117	78

>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5284]: Failed password for invalid user desktop from 87.194.216.51 port 3758 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3585]: Failed password for invalid user trac from 87.194.216.51 port 1852 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[4215]: Failed password for invalid user administrator from 87.194.216.51 port 2961 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[3845]: Failed password for invalid user trac from 87.194.216.51 port 2264 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[2524]: Failed password for invalid user art from 87.194.216.51 port 1067 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[5445]: Failed password for invalid user itmuser from 87.194.216.51 port 3867 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[1810]: Failed password for mail from 87.194.216.51 port 1769 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
>	07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[4476]: Failed password for invalid user ventrilo from 87.194.216.51 port 1627 ssh2 host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure

● 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.

```
source="tutorialdata.zip:*" "session opened"
| rex "for (?<raw_match>.+)"
| table _time, raw_match
```

Analizziamo il codice ancora una volta

source="tutorialdata.zip:*" "session opened"

> Cerca nei file contenuti dentro `tutorialdata.zip` tutte le linee che contengono la stringa **"Failed password"** (tipico messaggio dei log SSH quando una password è sbagliata).

| rex "for (?<raw_match>.+)"

> Usa rex (regular expression extractor) per catturare tutto quello che viene dopo la parola "for".
> Il risultato viene salvato in un nuovo campo chiamato raw_match.

| table _time, raw_match

Mostra i risultati in formato tabella con due colonne:

> _time → data e ora in cui la sessione è stata aperta
> raw_match → testo catturato dalla regex, cioè il dettaglio sull’utente/sessione

2025-08-07 10:50:38	user root by myuan(uid=0)
2025-08-07 10:50:38	user root by djohnson(uid=0)
2025-08-07 10:50:38	user djohnson by (uid=0)
2025-08-07 10:50:38	user nsharpe by (uid=0)
2025-08-07 10:50:38	user nsharpe by (uid=0)
2025-08-07 10:50:38	user djohnson by (uid=0)
2025-08-07 10:50:38	user myuan by (uid=0)
2025-08-07 10:50:38	user djohnson by (uid=0)
2025-08-07 10:50:38	user djohnson by (uid=0)

07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[98295]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[57580]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[82383]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[9114]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[86689]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[45574]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[29757]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[52801]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure
07/08/25 10:50:38,000	Mon Aug 07 2025 10:50:38 mailsv1 sshd[85151]: pam_unix(sshd:session): session opened for user djohnson by (uid=0) host = Javier source = tutorialdata.zip:.\mailsv/secure.log sourcetype = www1/secure

Dopo aver analizzato diversi log non ho trovato niente di particolare.

Crea una query Splunk per trovare tutti gli Internal Server Error.

source="tutorialdata.zip:*" host="Javier" status>=500 status<600

Analizziamo l'ultimo codice

source="tutorialdata.zip:*" host="Javier"

Cerca nei dati provenienti dal file tutorialdata.zip (tutti i file al suo interno *)
Solo gli eventi con host = Javier

Status>=500 status<600

mostra come i risultati i logs che hanno uno status tra 500 e 600 che sono quelli degli errori interni

10/08/25 18:18:59,000	198.35.1.75 - - [10/Aug/2025:18:18:59] "GET /cart.do?action=addtocart&itemId=EST-13&JSESSIONID=SD10SL2FF4ADFF53099 HTTP 1.1" 500 2324 "http://www.buttercupgames.com/category.screen?categoryId=NULL" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 645 host = Javier source = tutorialdata.zip:.\www1/access.log sourcetype = access_combined_wcookie
10/08/25 18:18:59,000	198.35.1.75 - - [10/Aug/2025:18:18:59] "GET /cart.do?action=addtocart&itemId=EST-13&JSESSIONID=SD10SL2FF4ADFF53099 HTTP 1.1" 500 2324 "http://www.buttercupgames.com/category.screen?categoryId=NULL" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 645 host = Javier source = tutorialdata.zip:.\www1/access.log sourcetype = access_combined_wcookie
10/08/25 18:18:59,000	198.35.1.75 - - [10/Aug/2025:18:18:59] "GET /cart.do?action=addtocart&itemId=EST-13&JSESSIONID=SD10SL2FF4ADFF53099 HTTP 1.1" 500 2324 "http://www.buttercupgames.com/category.screen?categoryId=NULL" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 645 host = Javier source = tutorialdata.zip:.\www1/access.log sourcetype = access_combined_wcookie
10/08/25 18:18:59,000	198.35.1.75 - - [10/Aug/2025:18:18:59] "GET /cart.do?action=addtocart&itemId=EST-13&JSESSIONID=SD10SL2FF4ADFF53099 HTTP 1.1" 500 2324 "http://www.buttercupgames.com/category.screen?categoryId=NULL" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 645 host = Javier source = tutorialdata.zip:.\www1/access.log sourcetype = access_combined_wcookie
10/08/25 18:18:55,000	198.35.1.75 - - [10/Aug/2025:18:18:55] "GET /product.screen?productId=SF-BVS-G01&JSESSIONID=SD10SL2FF4ADFF53099 HTTP 1.1" 500 2809 "http://www.buttercupgames.com/cart.do?action=view&itemId=EST-14" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 370 host = Javier source = tutorialdata.zip:.\www1/access.log sourcetype = access_combined_wcookie

Conclusione

In conclusione, Splunk si conferma uno strumento potente per raccogliere, analizzare e visualizzare grandi quantità di dati provenienti da sistemi diversi. L’uso delle query permette di filtrare le informazioni, individuare anomalie come errori interni o eventi sospetti e ottenere insight utili per il monitoraggio e la sicurezza dei sistemi. Anche se inizialmente la sintassi può sembrare complessa, con pratica e organizzazione delle query è possibile automatizzare le analisi e rendere la gestione dei dati più efficace e veloce.