Jon Taylor

Web Analytics and Security

Security Analytics Project SPL Queries

1. Brute Force Attacks

Which IPs attempted the most failed logins?

```
index=securecorp_logs event_type="login_failure"

| stats count by source_ip

| sort -count
```

The IP address 203.0.113.50 had the most failed login attempts, and was actually the only IP address to have a failed login attempt.

Which usernames were targeted the most?

```
index=securecorp_logs event_type= login_failure

| stats count by user

| sort -count
```

The user that had the most failed logins was jsmith.

Was your Splunk Username targeted?

```
index=securecorp_logs event_type="login_failure" user="taylorjl14"
```

My username was not targeted.

2. SQL Injection Attempts

Identify IPs attempting SQL Injection and the payloads used.

```
index=securecorp_logs event_type="sql_injection"

| table timestamp, source_ip, message
```

The IP 10.20.30.40 and 172.16.0.99 were attempting SQL Injections, and they were using the payload “ ‘ OR ‘1’=’1’ -- ". This is an attempt to bypass authentication utilizing user input. This is a classic SQL injection payload:

The ‘ closes the input

The “OR ‘1’=’1’ is always true, so it tricks the logic of the login query to think conditions are met for the login attempt.

The “- -“ comments, so the database ignores the rest of the query, so the hacker can login without a password.

Vulnerable login queries will fall for this injection, so it is imperative to write sound code for the website.

```
index=securecorp_logs event_type="sql_injection"

| table timestamp, source_ip, message

| stats count by source_ip

| sort -count
```

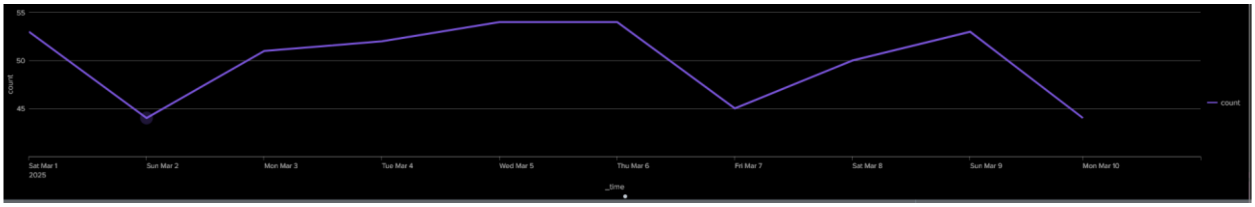
The above query provides a table with a count of how many times each IP tried to attempt a SQL Injection. 10.20.30.40 attempted 263 times and 172.16.0.99 attempted 237 times, indicating that there was somewhat equal pressure coming from both of them.

How frequently did SQL Injection attempts occur?

```
index=securecorp_logs event_type="sql_injection"

| timechart span=1d count
```

The above query creates a table indicating how many SQL Injections occurred per day in the data set. There seemed to be constant attempts, and the website only had three days in which there were under 50 attempts, March 2nd, March 7th , and March 10th.



3. Unauthorized Access

Which users attempted unauthorized access?

```
index=securecorp_logs event_type="unauthorized_access"

| stats count by user

| sort -count
```

The user “unknown” attempted unauthorized access. This indicates that this is NOT someone in our organization. Seeing unknown user is an immediate red flag, and this kind of login should be investigated further. There is a possibility that a hacker was testing multiple usernames or skipping the typical login steps. This could also mean that there is a misconfiguration in the system, so it’s also possible that it wasn’t malicious at all.

Did you see any privilege escalation attempts?

```
index=securecorp_logs event_type="privilege_escalation"

| table timestamp, user, source_ip, destination_ip, message
```

There were 163 privilege escalation attempts by the user “unknown”. This is a similar red flag, and could connect to the unknown user attempting to get access. Someone was trying to bypass the typical flow of the login process through vulnerabilities and elevate themselves to an admin, which likely means they are trying to attack the website altogether in a targeted way.

4. Malware Detections

Identify instances of malware detection. Who was affected?

```
index=securecorp_logs event_type="malware_detected"

| table timestamp, user, source_ip, severity, message

| sort -timestamp
```

There were 163 instances of malware detected, and the user “unknown’ was affected by the malware. The fact that the last three queries show the exact same number of 163 is very significant and probably indicates this is a connected, coordinated attack. The flow of the attack is concerning, considering they could have explored the site to find a vulnerability and then escalated their privileges to drop malware into the site.

5. Data Exfiltration

Find IPs that transferred the most data.

Check if there were large outbound transfers (>5GBs).

```
index=securecorp_logs

| stats sum(bytes_transferred) as total_bytes by source_ip

| sort -total_bytes
```

The IP 10.20.30.40 transferred the most data and the total amount of data transferred by that IP was 7.43 GB. This IP address is almost definitely the attacker as it is the same one that attempted SQL injection 263 times. I am lead to believe they were successful and there is a large threat to this website.

6. Firewall Blocks

Which ports were blocked the most? *Were there any denied RDP or SSH attempts?*

```
index=securecorp_logs action="denied"
| stats count by port
| sort -count
```

The port 8080 was blocked the most with 810 blocks, but ports 22, 80, 3389, 443, and 63 has similar amounts of blocks. Ports 22 and 3389 being blocked is significant because it indicates that there are denied SSH and RDP attempts. This means that someone is attempting to find access to the network remotely.

7. Bot Activity

Identify high-volume bot traffic based on the user_agent.

```
index=securecorp_logs
| stats count by user_agent
| sort -count
```

There were 1479 bot events. This indicates a high level of bot activity, as it is comparable to the traffic of normal web browsers such as Firefox and Chrome. There were also a large amount of events tied to HackerScanner, which is likely a program that scans for vulnerabilities.

Which IPs were scraping the website?

```
index=securecorp_logs
| stats count by visitor_ip, user_agent
| sort -count
```

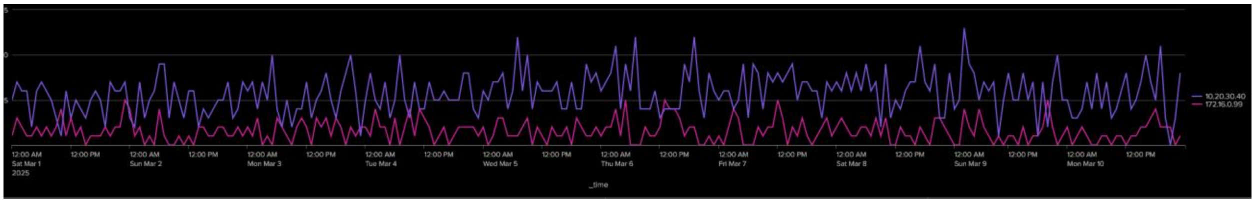
The IP address of 172.16.0.99 was scraping the website using a HackerScanner. This is significant because 172.16.0.99 was one of the IP addresses that attempted SQL injections. This could be the same attacker or a different attacker. I'd consider it likely that this is connected to the other IP address in the same coordinated attack on the website.

```
index=securecorp_logs (source_ip="10.20.30.40" OR source_ip="172.16.0.99")
| stats count by source_ip, event_type
```

Using this query, I found out that both IP addresses utilized similar attack methods.

```
index=securecorp_logs (source_ip="10.20.30.40" OR source_ip="172.16.0.99")
| timechart span=1h count by source_ip
```

This query displays that these attacks were ongoing at the same time from both IPs.



```
index=securecorp_logs (source_ip="10.20.30.40" OR source_ip="172.16.0.99")

| stats count by source_ip, port
```

This query displayed that IP 10.20.30.40 was on port 443 and 172.16.0.99 was on ports 22 and 3389. IP 10.20.30.40 was a web-based attack and 172.16.0.99 was tied to the network and web-based attack, indicating an additional remote approach. These could be different approaches from different hackers or a single group attempting to utilize multiple approaches to hack the website.