Official Incident Report

Event ID: 235

Rule Name: SOC127 - SQL Injection Detected

Table of contents

Alert

The alert was triggered due to various SQL injection attempts from the China-located 118[.]194.247.28 IP towards the system. The request that caused the alert to occur is shared below.

Request : GET /?douj=3034%20AND%201%3D1%20UNION%20ALL%20SELECT%201%2CNULL%2C%27%3Cscript%3Ealert%28%22XSS%22%29%3C%2Fscript%3E%27%2Ctable_name%20FROM%20information_schema.tables%20WHERE%202%3E1--%2F%2A%2A%2F%3B%20EXEC%20xp_cmdshell%28%27cat%20..%2F..%2F..%2Fetc%2Fpasswd%27%29%23

EventID :	235
Event Time :	Mar, 07, 2024, 12:51 PM
Rule :	SOC127 - SQL Injection Detected
Level :	Security Analyst
Source Address :	118.194.247.28
Destination Address :	172.16.20.12
Destination Hostname :	WebServer1000
Request URL :	GET /?douj=3034%20AND%201%3D1%20UNION%20ALL%20SELECT%201%2CNULL%2C%27%3Cscript%3Ealert%28%22XSS%22%29%3C%2Fscript%3E%27%2Ctable_name%20FROM%20information_schema.tables%20WHERE%202%3E1--%2F%2A%2A%2F%3B%20EXEC%20xp_cmdshell%28%27cat%20..%2F..%2F..%2Fetc%2Fpasswd%27%29%23 HTTP/1.1 200 865
Device Action :	Allowed
Show Hint	

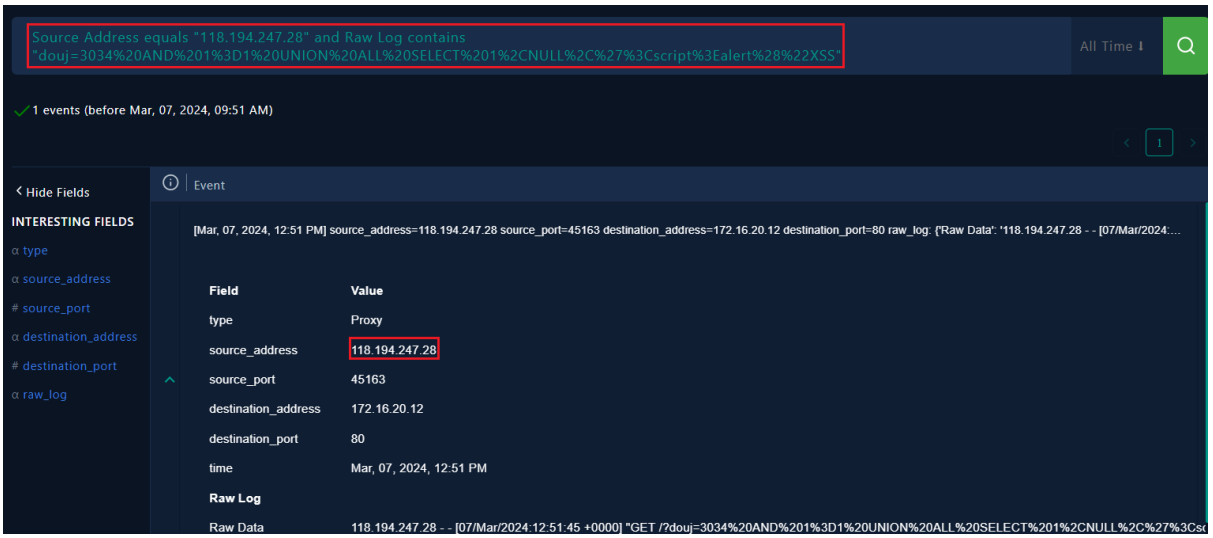
First, the alert should be verified by checking the available logs, and then it should be determined whether the attack was successful or not.

Detection

Verify

You can search for the attacker IP on Log Management to have a better understanding of the alert. This search result shows proxy and firewall logs from different years. You should examine all requests coming from IP 118[.]194.247.28 to confirm the alert.

Search for the logs of the attacker IP to confirm the alert as shown below. The request that triggered the alert was seen in raw data. Thus, it can be said that the alert is True Positive.



Analysis

Reputation Check

You should check the reputation of the IP "118[.]194.247.28" which caused the alert to be triggered.

118.194.247.28 was found in our database!

This IP was reported **4,070** times. Confidence of Abuse is **100%**: ?

100%

ISP	Beijing CNISP Technology Co. Ltd.
Usage Type	Fixed Line ISP
Domain Name	cnisp.org.cn
Country	China
City	Beijing, Beijing

IP info including ISP, Usage Type, and Location provided by IP2Location. Updated monthly.

REPORT 118.194.247.28 | WHOIS 118.194.247.28

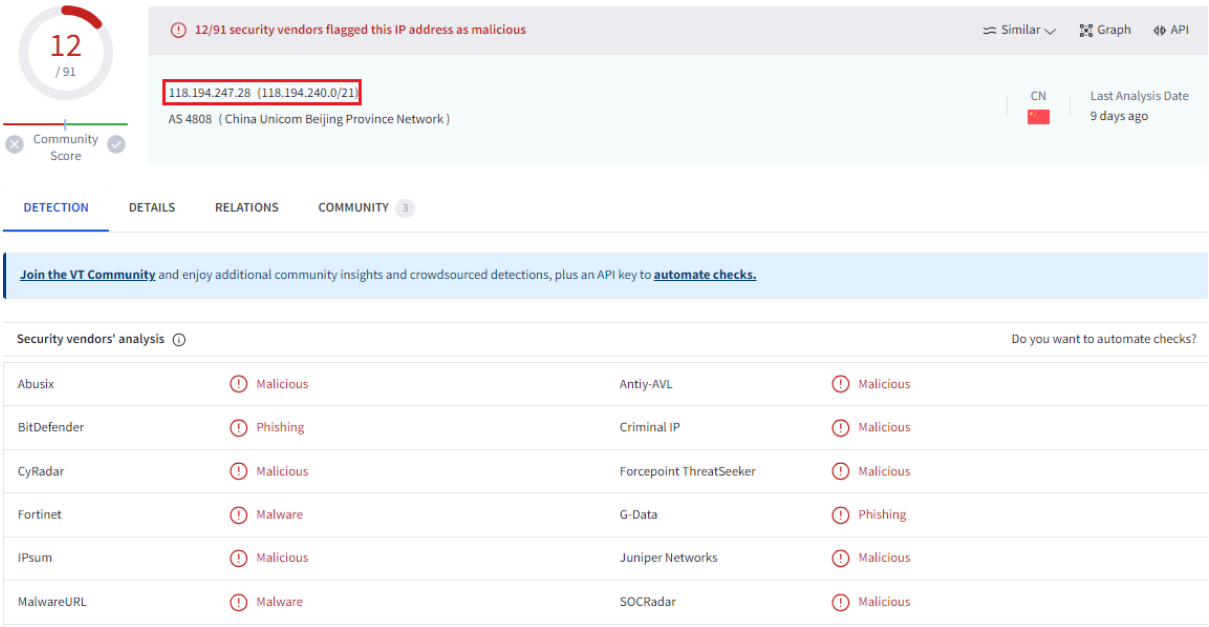
IP Abuse Reports for 118.194.247.28:

This IP address has been reported a total of **4,070** times from 484 distinct sources. 118.194.247.28 was first reported on June 10th 2022, and the most recent report was **4 hours ago**.

Recent Reports: We have received reports of abusive activity from this IP address within the last week. It is potentially still actively engaged in abusive activities.

Reporter	IoA Timestamp	Comment	Categories
jioni.de	2024-03-11 03:41:34 (4 hours ago)	2024-03-10 19:23:03,230 INFO [ImapServer-2601] [ip=95.216.27.198;oip=::ffff:118.194.247.28;via=::ff ... show more	Brute-Force
Axel	2024-03-11 00:15:01 (7 hours ago)	This IP was banned by Fail2Ban on behalf of 26ThAve. Reason: Multiple incorrect SSH login credential ... show more	SSH
opcenter	2024-03-10 22:29:56 (9 hours ago)	2024-03-10 dovecot_login authenticator failed for ([118.194.247.28]) [118.194.247.28]: 535 Incorrect ... show more	Brute-Force
cticom.ms	2024-03-10 20:44:08 (11 hours ago)	Email Auth Brute force attack 3/3 in last day	Brute-Force
Georgie	2024-03-10 19:25:11 (12 hours ago)	...	Brute-Force

hxxps://www.abuseipdb.com/check/118.194.247.28



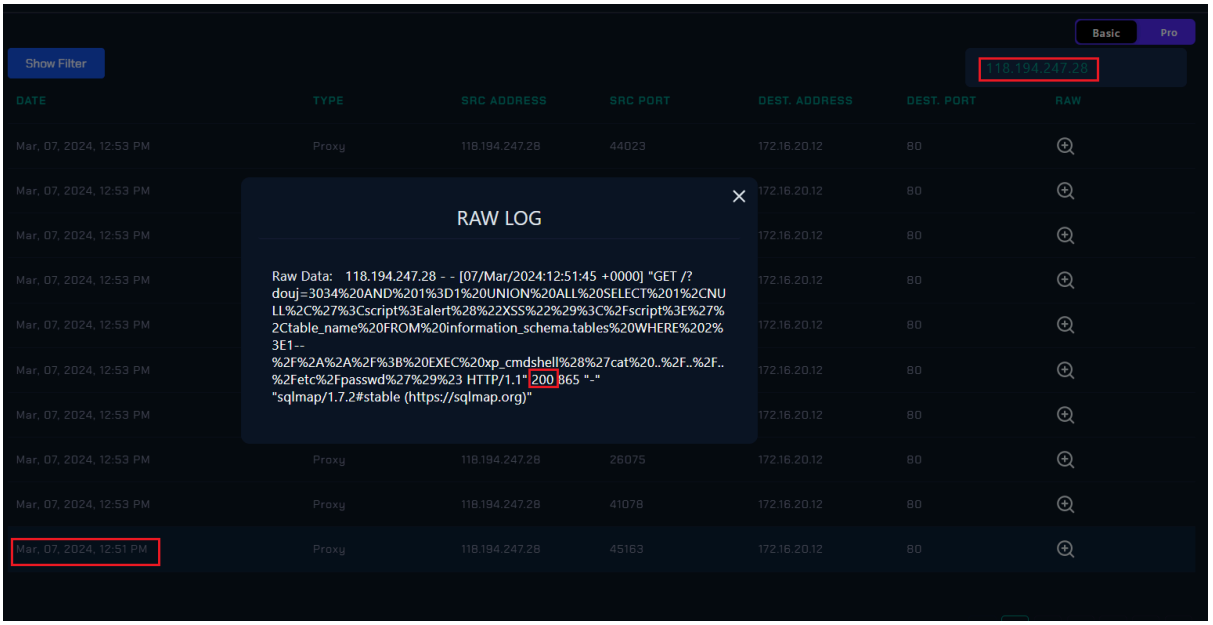
hxxps://www.virustotal.com/gui/ip-address/118.194.247.28

The related IP was reported as malicious, phishing, web attack, and brute force in sources such as Virus Total and AbuseIPDB.

When all logs of the attacker IP are searched on Log Management, it is seen that requests from many different ports were seen towards the system before the alert. A request to port 80 was also seen at 12:50 PM as a result of port scan traffic. Subsequently, proxy logs were detected. Thus, it is thought that the attacker first performed a port scan on the system and understood that port 80 was open to remote. Then they performed SQL injection attempts.

When the logs in the proxy were examined in detail, first, the following request was considered suspicious. It received http response code 200(ok) after the request.

hxxps://developer.mozilla.org/en-US/docs/Web/HTTP/Status/200



Request :
/?douj=3034%20AND%201%3D1%20UNION%20ALL%20SELECT%201%2CNULL%2C%27%3Cscript%3Ealert%28%22XSS%22%29%3C%2Fscript%3E%27%2Ctable_name%20FROM%20information_schema.tables%20WHERE%202%3E1--%2F%2A%2A%2F%3B%20EXEC%20xp_cmdshell%28%27cat%20..%2F..%2F..%2Fetc%2Fpasswd%27%29%23

This request appears to be an SQL injection attack on the douj parameter. The request appears to have been made by sqlmap, a tool that performs an SQL injection attack. It appears that UNION ALL SELECT was used to retrieve the information of the requested database tables, followed by a command call using xp_cmdshell (cat ../../etc/passwd). This is an attack attempt to pull the contents of the /etc/passwd file on the server.

sqlmap : sqlmap is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over database servers.

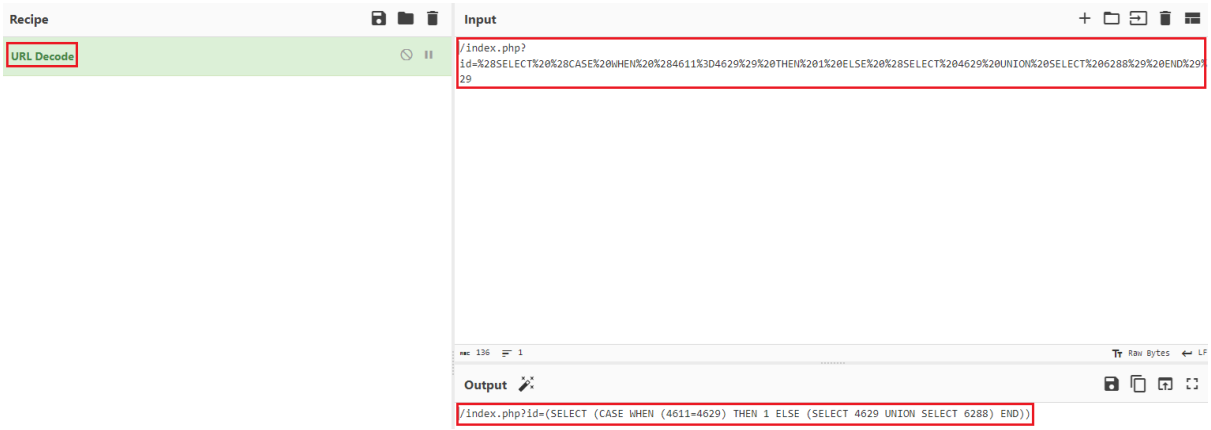
In addition, a few different requests are examined and shared below.

The first log of the related IP is seen in the proxy. It is seen in the relevant log that the request sent to the address hosted at IP 172[.]16.20.12 received 200 (success status response code) on the proxy. This means that the related host is open to remote access.

Request:
/index.php?id=%28SELECT%20%28CASE%20WHEN%20%284611%3D4629%29%20THEN%201%20ELSE%20%28SELECT%204629%20UNION%20SELECT%206288%29%20END%29%29

When the relevant request is decoded in CyberChef, the following result is obtained.

Output: /index.php?id=(SELECT (CASE WHEN (4611=4629) THEN 1 ELSE (SELECT 4629 UNION SELECT 6288) END))



This output contains an inner query that must be processed as part of an SQL query. The CASE WHEN section returns a value depending on a condition. If the number 4611 is equal to 4629, a value of 1 is returned. Otherwise, a subquery containing the numbers 4629 and 6288 is returned.

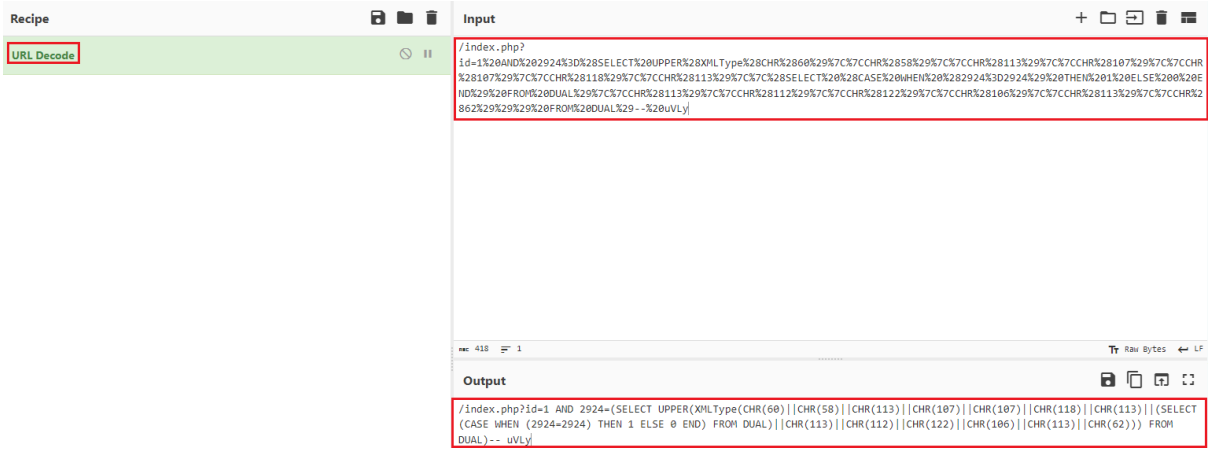
Request:
/index.php?id=1%20AND%20EXTRACTVALUE%287321%2CCONCAT%280x5c%2C0x716b6b7671%2C%28SELECT%20%28ELT%287321%3D7321%2C1%29%29%29%2C0x71707a6a71%29%29

Output: /index.php?id=1 AND EXTRACTVALUE(7321,CONCAT(0x5c,0x716b6b7671,(SELECT (ELT(7321=7321,1))),0x71707a6a71))

EXTRACTVALUE(7321, CONCAT(0x5c,0x716b6b7671,(SELECT (ELT(7321=7321,1))),0x71707a6a71)): This section is an SQL statement that tries to extract a value from an XML string. The expression SELECT (ELT(7321=7321,1)) is a conditional statement of SQL. If 7321 is equal to 7321, it will return 1. The nested query will return 1, since this condition is always true here. Ultimately, it aims to get the value 0x716b6b7671 from the string.

Request:
/index.php?id=1%20AND%202924%3D%28SELECT%20UPPER%28XMLType%28CHR%2860%29%7C%7CCHR%2858%29%7C%7CCHR%28113%29%7C%7CCHR%28107%29%7C%7CCHR%28107%29%7C%7CCHR%28118%29%7C%7CCHR%28113%29%7C%7C%28SELECT%20%28CASE%20WHEN%20%282924%3D2924%29%20THEN%201%20ELSE%200%20END%29%20FROM%20DUAL%29%7C%7CCHR%28113%29%7C%7CCHR%28112%29%7C%7CCHR%28122%29%7C%7CCHR%28106%29%7C%7CCHR%28113%29%7C%7CCHR%2862%29%29%29%20FROM%20DUAL%29--%20uVLy

Output: /index.php?id=1 AND 2924=(SELECT UPPER(XMLType(CHR(60)||CHR(58)||CHR(113)||CHR(107)||CHR(107)||CHR(118)||CHR(113)||(SELECT (CASE WHEN (2924=2924) THEN 1 ELSE 0 END) FROM DUAL)||CHR(113)||CHR(112)||CHR(122)||CHR(106)||CHR(113)||CHR(62))) FROM DUAL)-- uVLy



This URL probably represents an SQL injection attack. The code after the "index.php?id=" part of the URL could be part of an SQL query. The SQL query appears to be trying to extract a value from an XML string using the SELECT UPPER(XMLType(...)) function.

SELECT UPPER(XMLType(CHR(60)||CHR(58)||CHR(113)||CHR(107)||CHR(118)||CHR(113)||(SELECT (CASE WHEN (2924=2924) THEN 1 ELSE 0 END) FROM DUAL)||CHR(113)||CHR(112)||CHR(122)||CHR(106)||CHR(113)||CHR(62))) FROM DUAL: This section is an SQL statement that tries to extract a value from an XML string. It uses a nested query (subquery). If 2924 is equal to 2924, it returns 1. Otherwise, it returns 0. As a result, the XML string CHR(60)||CHR(58)||CHR(113)||CHR(107)||CHR(118)||CHR(113)||CHR(113)||CHR(1)||CHR(113)||CHR(112)||CHR(122)||CHR(106)||CHR(113)||CHR(62) was created and converted to upper case.

To summarize, the attacker used a hacking tool to send various malicious requests to the target system. Attackers who performed SQL injection attack can interact with the database of the affected web application and perform various malicious activities. Here are some of the information and malicious activities that attackers can obtain through SQL injection attack:

- **Access to Sensitive Data:** Attackers can access sensitive information in the database using SQL injection. This information can include usernames, passwords, credit card information, personal identification information, etc.
- **Data Deletion and Modification:** Attackers can delete or modify data in the database using SQL injection. This can corrupt or destroy records in the database, manipulate operational data, or insert misleading information.
- **Bypassing Authentication:** Attackers can bypass authentication mechanisms using SQL injection. This allows them to gain unauthorized access and log in to the system or gain more privileges.

- **Command Execution:** Attackers can execute commands directly on the database server using SQL injection. This can result in hijacking the server or attacking other network resources by executing malicious code on the database server.
- **Sensitivity Testing:** Attackers can perform sensitivity testing of the application using SQL injection. This can be used to discover vulnerabilities of the application and target weak points for further attacks.
- **Database Server Control:** Attackers can compromise the database server using SQL injection and perform unauthorized operations on it. This can lead to abuse of server resources and other types of attacks.

It is seen that the requests in the logs above returned the http response code 200. What does this mean? Did the attacker succeed in the SQL injection attack?

HTTP response code 200 indicates that the server successfully processed the request and sent a correct response to the client. However, receiving an HTTP response code 200 does not confirm that the SQL injection attack was successful. The SQL injection test only checks whether the server can redirect to the database. HTTP response code 200 shows that the server successfully processed the request and indicates whether this was caused by an SQL injection attack. Therefore, receiving HTTP response code 200 does not confirm that the attack was successful. It only indicates that the server successfully processed the request. In other words, the server's application (database) logs should be checked to see if the attack was successful. The requests may have come to the client but failed. For example, some databases such as MySQL return error messages due to invalid SQL queries. These error messages can help the attacker verify the attack target.

Lesson Learned

- Hosts should not be opened to remote or unauthorized users unless necessary, even in test environments.
- When the hosts open to remote have authentication structures, precautions should be taken against Brute Force attacks in the system. For instance, MFA or recaptcha structure should be activated.
- The structures open to remote must be up-to-date in order not to be affected by vulnerabilities.
- In structures open to remote, various security products should be used to detect malware and protect the system against Web attacks and their signatures/rules must be up to date.
- Input and output traffic should be monitored, potential SQL injection attacks should be detected and the system should be protected by using security tools such as firewalls and Intrusion Prevention Systems (IPS).
- Developers and system administrators should be trained against SQL injection (Web attack) attacks. This training will help them understand how attacks happen and how to prevent them.
- Web applications should be tested for security regularly. These tests identify potential vulnerabilities and identify areas that need to be fixed.

Appendix

MITRE

MITRE Tactics	MITRE Techniques
Reconnaissance	• Active Scanning: Vulnerability Scanning
Initial Access	• Exploit Public-Facing Application
Credential Access	• Unsecured Credentials: Credentials In Files

Artifacts

Field	Value
IPs	• 172[.]16.20.12 • 118[.]194.247.28
Host	• WebServer