## **Incident Overview**

On **March 7, 2024**, an internal web server (**WebServer1000 / Atlanta‑Server**) was targeted by an **automated SQL injection exploitation campaign** originating from an external IP address (**118.194.247.28**). The activity was detected by the **SOC127 – SQL Injection Detected** rule and occurred over a **two‑minute window (12:51–12:53 PM)**. The attacker used **sqlmap**, a well‑known offensive security tool, to systematically probe, confirm, and escalate database injection vulnerabilities.

## **Affected Assets**

* **Destination Host:** WebServer1000 (Atlanta‑Server)
* **IP Address:** 172.16.20.12
* **Operating System:** Ubuntu 20.04.02 (64‑bit)
* **Role:** Internal web application server
* **Last User Logon:** Nov 10, 2023

## **Source & Threat Intelligence**

* **Source IP:** 118.194.247.28
* **Geolocation:** China
* **Ownership:** Beijing CNISP Technology Co., Ltd.
* **Reputation:**
  + Flagged malicious by 8 **security vendors**
  + Associated with scanning and exploitation activity
* **Tooling Identified:** sqlmap/1.7.2#stable (via User‑Agent)

The source IP is externally owned, has no legitimate business relationship with the organization, and demonstrates behavior consistent with **automated exploitation frameworks**.

## **Attack Timeline**

All malicious activity occurred within a **short, high‑intensity burst**, consistent with automated tooling:

* **12:51 PM:** Initial boolean‑based and UNION‑based SQL injection tests
* **12:52 PM:** Error‑based SQL injection and parser fuzzing payloads
* **12:53 PM:** Advanced exploitation attempts, including stacked queries and command execution payloads

The attacker rotated source ports rapidly while maintaining a consistent target.

## **Attack Vector & Techniques**

**Primary Attack Vector:** SQL Injection via HTTP GET parameters

**Techniques Observed:**

* Boolean‑based SQL injection (AND 1=1)
* UNION‑based SQL injection for column and schema enumeration
* Error‑based SQL injection using CAST, CASE, and CHR() functions
* SQL syntax fuzzing to identify backend parser behavior
* Cross‑Site Scripting (XSS) payload injection
* Stacked queries attempting **OS command execution** via xp\_cmdshell

These techniques demonstrate a **full exploitation lifecycle**, progressing from vulnerability discovery to attempted system‑level access.

## **Evidence of Exploitation Attempt**

* Multiple crafted SQL payloads targeting the same parameters
* Consistent **HTTP 200 responses** with identical response sizes (865 bytes)
* Direct attempts to read system files (/etc/passwd)
* Malicious User‑Agent explicitly identifying sqlmap
* High‑confidence attack signatures matched across multiple logs

Although no definitive evidence of data exfiltration was observed, application behavior indicates the backend **processed injected input**, confirming exploitability.

## **Impact Assessment**

* **Confidentiality:** At risk due to schema enumeration and data extraction attempts
* **Integrity:** Potential compromise through SQL‑level manipulation
* **Availability:** No denial‑of‑service observed
* **Overall Risk:** **High**, due to confirmed injection handling and escalation attempts

## **Detection & Classification**

* **Detection Source:** Proxy / Web logs
* **Detection Rule:** SOC127 – SQL Injection Detected
* **Classification:** **True Positive**
* **Severity:** High

This activity represents a **deliberate, malicious exploitation attempt**, not false positives or benign scanning.

## **Response & Recommendations**

**Immediate Actions:**

* Block source IP at perimeter controls
* Review application logs for signs of successful data access
* Validate database permissions (disable stacked queries and xp\_cmdshell)

**Long‑Term Mitigations:**

* Implement parameterized queries / prepared statements
* Deploy a Web Application Firewall (WAF) with SQLi rules
* Apply strict input validation and output encoding
* Conduct secure code review and penetration testing

## **Analyst Conclusion**

This incident demonstrates a **real‑world, automated SQL injection attack** leveraging industry‑standard offensive tooling against a production web server. The attacker exhibited clear intent to escalate from database access to operating system command execution. Rapid detection and analysis confirmed the **true positive nature** of the alert and highlighted critical areas for application hardening.