User Agent

A User Agent is **a string sent in HTTP requests by clients (e.g., browsers, bots, or scripts) to identify themselves to servers**. The User Agent string includes details about the client's software, operating system, and sometimes the version number. **Attackers often spoof User Agents** to disguise their activities, making it critical to analyze User Agents to determine if requests are coming from legitimate browsers or potentially malicious botnets.

1. What Is a User Agent?

The User-Agent header is **included in HTTP requests**.

Example User-Agent for a Legitimate Browser

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/115.0.5790.171 Safari/537.36
```

Breakdown

- Mozilla/5.0: Historical prefix for compatibility.
- Windows NT 10.0; Win64; x64: Operating system and architecture.
- AppleWebKit/537.36: Browser engine.
- Chrome/115.0.5790.171: Browser name and version.
- Safari/537.36: Secondary engine for compatibility.

2. How User Agents Are Used

- **Identify Browsers**
  - Helps servers optimize content for specific browsers and devices.
- **Detect Bots**
  - Some bots or crawlers have unique User-Agent strings (e.g., search engines like Googlebot).
- **Filter Malicious Requests**
  - User-Agent analysis helps identify botnets or malicious scripts.
- **Logging and Analytics**
  - Web logs rely on User-Agent strings to analyze traffic patterns.

3. Legitimate Browser vs. Botnet

a. Characteristics of Legitimate User Agents

1. Valid Format

- Legitimate browsers follow standardized User-Agent formats.

2. Matching Behavior

- The User-Agent matches typical browser behavior (e.g., valid headers, cookies, and rendering requests).

3. Consistent Traffic

- Human-driven browsing patterns include clicks, delays, and varied URLs.

4. Common Strings

- Well-known browser strings like Chrome, Firefox, Safari, or Edge.

b. Characteristics of Suspicious User Agents (Botnets or Scripts)

1. Missing or Generic Strings

- Bots often use simplified or malformed User-Agent strings.
- Example

```
User-Agent: Python-urllib/3.7
```

2. Outdated or Nonexistent Browsers

- Requests claim to use very old or invalid browser versions.
- Example

```
User-Agent: Mozilla/3.0 (compatible; Win95)
```

3. Impersonating Legitimate Browsers

- Bots mimic legitimate User-Agent strings but behave suspiciously (e.g., high request volume).

4. Rapid Traffic

- Bots generate large numbers of requests in short timeframes (automation).

5. Missing Headers

- Bots often send minimal or missing HTTP headers (e.g., no cookies, referrers).

4. Tools to Analyze User Agents

1. Manual Analysis

- Use tools like curl or browser dev tools to inspect User-Agent headers.

Example curl command

```
curl -A "User-Agent: Mozilla/5.0" https://example.com
```

2. User Agent Parsers

- Online tools to analyze and decode User-Agent strings
  - WhatIsMyBrowser.com
  - UserAgentString.com

3. Log Analysis Tools

- Use log analysis tools like ELK Stack (Elasticsearch, Logstash, Kibana) to analyze User-Agent strings in HTTP logs.

4. Web Application Firewalls (WAF)

- Tools like ModSecurity, Cloudflare, or AWS WAF inspect User-Agents and block malicious ones.

5. Threat Intelligence Feeds

- Use updated botnet and crawler User-Agent blacklists.

5. Detecting Botnets or Malicious Clients

a. Identify Known Bots

- Legitimate bots (e.g., Googlebot) announce themselves clearly.
- Verify IP ranges and perform reverse DNS lookups for confirmation.

b. Check for Anomalies

- Compare User-Agent behavior to legitimate patterns
- Does it lack cookies or referrer headers?
- Does it send a flood of requests in rapid succession?

c. User-Agent Blacklists

- **Maintain a list of known bad User-Agent strings**.
- Example of suspicious User-Agents

```
User-Agent: curl/7.61.1
User-Agent: Python-requests/2.25.1
User-Agent: Mozilla/5.0 (compatible; Bot/1.0)
```

d. Behavior-Based Detection

- Look beyond the User-Agent string and **analyze behavior**
- Traffic patterns: High-frequency requests from the same IP.
- Header consistency: Legitimate browsers typically include headers like Accept, Accept-Encoding, and Cookie.

6. Example of Legitimate vs. Malicious Traffic

Aspect	Legitimate Browser	Botnet/Script
User-Agent String	Properly formatted (Chrome, Firefox, Safari).	Simplified, missing, or malformed.
Behavior	Human-like clicks and delays.	Rapid-fire requests with no delays.
HTTP Headers	Includes cookies, referrer, and accept headers.	Minimal headers, no cookies or referrers.
Traffic Volume	Low and varied.	High and repetitive.
IP Range	Distributed, user-based IPs.	Concentrated, unusual IPs.

7. Best Practices to Mitigate Malicious Bots

1. **Validate User-Agent Strings**

- Use whitelists for known legitimate browsers.

2. **Implement Rate Limiting**

- Restrict the number of requests per IP in a set timeframe.

3. **Use a Web Application Firewall (WAF)**

- Inspect and block suspicious User-Agent strings.

4. **Behavioral Analysis**

- Monitor traffic patterns to identify automation.

5. **CAPTCHA Challenges**

- Introduce CAPTCHAs for suspicious behavior to confirm human users.

6. **Threat Intelligence**

- Update blacklists regularly to block known bad User-Agent strings.

8. Summary

Aspect	Details
What is a User Agent?	A string that identifies the client (browser, bot, or script) to the server.
Legitimate Browsers	Use well-formatted, up-to-date User-Agent strings.
Botnet/Script Indicators	Simplified strings, high traffic, missing headers, or outdated versions.
Detection Methods	Manual analysis, log inspection, User-Agent parsing tools, WAF.
Mitigation	Validate strings, rate limit, implement behavioral analysis, use CAPTCHAs.

While User-Agent strings are useful for identifying clients, they can be **easily spoofed**. To reliably differentiate legitimate browsers from botnets or scripts, **combine User-Agent validation with behavior-**

based analysis, rate limiting, and tools like web application firewalls. Regularly updating detection techniques and leveraging threat intelligence ensures your systems stay protected.