#! Anatomy of a Bug

Autopsy of CVSS 9.8

The Fortinet 0days

The Patient

- **Name:** Fortinet FortiWeb
- **CVE:** CVE-2025-64446, CVE-2025-58034
- **Diagnosis:** Full Authentication Collapse
- **Vector:** fwbcgi
- **Severity:** 🔥 9.8

What does 9.8 mean?

- **Auth:** None
- **User Interaction:** None
- **Attack Vector:** Remote
- **Result:** Root-level command execution

#1 The Context

The Appliance Illusion

#1.1 The Context

The Appliance Illusion

👉 FortiWeb is positioned as a Web Application Firewall, guarding critical assets.

👉 Internally, it relies on Apache + CGI binaries, a model dating back decades.

👉 Security assumptions:

👉 Requests reaching CGI are trusted.

👉 Headers originate from internal components.

👉 These assumptions are invalid on an internet-facing device.

#2 The Architecture

CGI and fwbcgi

#2.1 The Architecture

CGI and fwbcgi

👉 The web server routes requests to fwbcgi, the administrative CGI binary.

👉 Routing is defined via httpd.conf using ScriptAlias.

👉 fwbcgi performs:
- 👉 Input validation (cgi_inputcheck)
- 👉 Authentication (cgi_auth)
- 👉 Command dispatch (cgi_process)

👉 The security model depends on these checks executing in the correct order.

#3 The Vulnerability

CVE-2025-64446

#3.1 The Vulnerability (1/2)

Path Resolution Before Trust

👉 Legitimate API endpoints live under: /api/v2.0/cmdb/...

👉 Apache processes the URL path before enforcing logical boundaries.

👉 Traversal sequences (../) were not normalized.

👉 Result: An attacker can escape the API path and directly invoke fwbcgi.

#3.2 The Vulnerability (1/2)

The Authentication Mirage

👉 fwbcgi calls cgi_auth() expecting internal IPC context.

👉 Authentication inferred via header: CGIINFO.

👉 Assumption: Only set by trusted internal sources.

👉 This assumption is catastrophically false.

#3.3 The Vulnerability (1/2)

Context Hydration via Header

👉 cgi_auth() logic:

👉 Base64-decodes CGIINFO.

👉 Parses JSON.

👉 Extracts username, profname, vdom, loginname.

👉 Passed to set_login_context_vsa().

👉 No cryptographic verification. No origin check.

👉 Result: Unauthenticated Super-Admin Access.

#4 The Vulnerability

CVE-2025-58034

#4.1 The Vulnerability (2/2)

Command Construction without Boundaries

👉 Next, the request executes in a fully hydrated admin context.

👉 Administrative requests are routed into policy_scripting_post_handler, a CGI handler responsible for managing policy automation scripts.

👉 This handler must invoke an underlying OS utility to:

- 👉 create scripts
- 👉 update scripts
- 👉 execute scripts

👉 To do so, it constructs a shell command at runtime.

#4.2 The Vulnerability (2/2)

The Critical Design Error

👉 The handler builds the command using string concatenation, not argument vectors.

👉 User-controlled JSON fields (e.g. name, script, or comment) are inserted verbatim.

👉 It passes the arguments to run_script:

```
/usr/local/bin/run_script --name <user_input>
```

#4.3 The Vulnerability (2/2)

Why This Is Catastrophic

👉 Common in appliance firmware,
the CGI process runs as root.

👉 There is:
- 👉 no privilege drop
- 👉 no chroot
- 👉 no seccomp

👉 Result: the injected command
executes as UID 0.

#5 The Kill-Chain

Step-by-Step Execution

#5.1 The Kill-Chain

Step 1: Path Traversal

👉 Escape API routing via ../ path traversal sequences.

👉 Bypass logical API boundaries enforced at the application layer.

👉 Reach the internal CGI binary fwbcgi directly, outside the authenticated API flow.

👉 Payload: POST /api/v2.0/cmdb/system/admin/../../../../../cgi-bin/fwbcgi

#5.2 The Kill-Chain

Step 2: Identity Forgery

👉 Attacker supplies:

👉 {
"username": "admin",
"profname": "super_admin",
"vdom": "root",
"loginname": "admin"
}

👉 Encoded.

👉 Injected into CGIINFO.

👉 fwbcgi decodes and parses the header.

👉 User session is hydrated as super_admin.

#5.3 The Kill-Chain

Step 3: OS Command Injection

👉 Payload: test; /bin/sh -c 'id'

👉 Shell interpretation:

👉 Execute legitimate command: /usr/local/bin/run_script --name test

👉 Then execute attacker command: /bin/sh -c 'id'

👉 The second command is not part of the application logic; it is executed solely by the shell as a consequence of string concatenation and shell parsing rules.

👉 Result: RCE as root.

#5.4 The Kill-Chain

Step 5: Post-Exploitation

👉 Following FortiWeb, attackers leveraged Cisco AsyncOS appliances.

👉 Exploited via CVE-2025-20393 to establish persistence and maintain control.

👉 Forensic analysis confirms FortiWeb foothold enabled downstream Cisco execution.

👉 Reference: #! Anatomy of a Bug #6

#6 The Fix.

Detailed Remediation

#6.1 The Fix.

Restricted Routing

👉 Deny direct access to /cgi-bin/ at the web-server layer.

👉 Collapse the attack surface by preventing external invocation of CGI binaries entirely.

👉 Restrict fwbcgi execution to internal-only redirects originating from authenticated handlers.

#6.2 The Fix.

Strict Authentication

👉 Remove header-based trust entirely.

👉 Eliminate implicit IPC assumptions between web components.

👉 Bind authentication state to cryptographically verified session context:

- 👉 Session identifier
- 👉 Cryptographic verification
- 👉 Server-side session lookup
- 👉 Explicit privilege mapping

#6.3 The Fix.

Safe Execution

👉 **Eliminate shell invocation from all request paths.**

👉 **Remove string-based command construction.**

👉 **Use strict argument vectors with no shell interpretation.**

#6.4 The Fix.

Safe Exection

```
// What existed before: (VERY DANGEROUS)
system("/usr/local/bin/run_script --name " + user_input);

// What must exist instead:
execve(
  "/usr/local/bin/run_script",
  ["run_script", "--name", user_input],
  env
);
```

#7 Developer's Takeaway

Trust is not a boundary

#7.1 Developer's Takeaway

Trust is not a boundary

👉 **Headers are user input, not a trust signal.**

👉 **CGI is not sandboxed and executes with ambient privilege.**

👉 **Perimeter devices must assume hostile traffic by default.**

👉 **This was not a logic failure; it was a systemic design failure.**

Status

Patched (FortiWeb 8.0.2+).
Know your sanitization.
Know your input.

#! Anatomy of a Bug

Technical Credits:

watchTowr, Fortinet PSIRT

Author: @tralsesec

#AOAB #AnatomyOfABug #ExploitDev
#Fortinet #WAF #Firewall #ZeroDay
#CVE202564446 #CVE202558034