Authentication bypassed with malformed request URI on nginx

Critical **clems4ever** published **GHSA-68wm-pfjf-wqp6** on May 28, 2021

Package

authelia

Affected versions

v4.0.0-alpha1 to v4.29.2

Patched versions

v4.29.3

Description

Impact

This affects uses who are using nginx ngx_http_auth_request_module with Authelia, it allows a malicious individual who crafts a malformed HTTP request to bypass the authentication mechanism. It additionally could theoretically affect other proxy servers, but all of the ones we officially support except nginx do not allow malformed URI paths.

Patches

The problem is rectified entirely in v4.29.3. As this patch is relatively straightforward we can back port this to any version upon request. Alternatively we are supplying a git patch to 4.25.1 which should be relatively straightforward to apply to any version, the git patches for specific versions can be found below.

▶ Patch for 4.25.1:

Workarounds

The most relevant workaround is upgrading. **If you need assistance with an upgrade please contact us on Matrix or Discord.** Please just let us know you're needing help upgrading to above 4.29.2.

You can add an block which fails requests that contains a malformed URI in the internal location block. We have crafted one that should work in most instances, it basically checks no chars that are required to be URL-encoded for either the path or the query are in the URI. Basically this regex checks that the characters between the square braces are the only characters in the $request_uri header, if they exist, it returns a HTTP 401 status code. The characters in the regex match are tested to not cause a parsing error that would result in a failure, however they are not exhaustive since query strings seem to not always conform to the RFC.

▶ authelia.conf:

Discovery

This issue was discovered by:

Siemens Energy
Cybersecurity Red Team

- Silas Francisco
- Ricardo Pesqueira

Identifying active exploitation of the vulnerability

The following regex should match log entries that are an indication of the vulnerability being exploited:

```
level=error msg="Unable to parse target URL: Unable to parse URL (extracted from X-Original-URL header)?.*?: parse.*?net/url:.*github\.com/authelia/authelia/internal/handlers/han
```

Example log entry ***with*** X-Original-URL configured:

```
time="2021-05-21T16:31:15+10:00" level=error msg="Unable to parse target URL: Unable to parse URL extracted from X-Original-URL header: parse \"https://example.com/": net/url:
invalid control character in URL" method=GET path=/api/verify remote_ip=192.168.1.10 stack="github.com/authelia/authelia/internal/middlewares/authelia_context.go:65
(*AutheliaCtx).Error\ngithub.com/authelia/authelia/internal/handlers/handler_verify.go:431
VerifyGet.func1\ngithub.com/authelia/authelia/internal/middlewares/authelia_context.go:50 AutheliaMiddleware.func1.1\ngithub.com/fasthttp/router@v1.3.12/router.go:414
(*Router).Handler\ngithub.com/authelia/authelia/internal/middlewares/log_request.go:14      LogRequestMiddleware.func1\ngithub.com/valyala/fasthttp@v1.24.0/server.go:2219
(*Server).serveConn\ngithub.com/valyala/fasthttp@v1.24.0/workerpool.go:223                    (*workerPool).workerFunc\ngithub.com/valyala/fasthttp@v1.24.0/workerpool.go:195
(*workerPool).getCh.func1\nruntime/asm_amd64.s:1371                                                  goexit"
```

Example log entry ***without*** X-Original-URL configured:

```
time="2021-05-21T16:30:17+10:00" level=error msg="Unable to parse target URL: Unable to parse URL https://example.com/: parse \"https://example.com/": net/url: invalid control
character in URL" method=GET path=/api/verify remote_ip=192.168.1.10 stack="github.com/authelia/authelia/internal/middlewares/authelia_context.go:65
(*AutheliaCtx).Error\ngithub.com/authelia/authelia/internal/handlers/handler_verify.go:431
VerifyGet.func1\ngithub.com/authelia/authelia/internal/middlewares/authelia_context.go:50 AutheliaMiddleware.func1.1\ngithub.com/fasthttp/router@v1.3.12/router.go:414
(*Router).Handler\ngithub.com/authelia/authelia/internal/middlewares/log_request.go:14      LogRequestMiddleware.func1\ngithub.com/valyala/fasthttp@v1.24.0/server.go:2219
(*Server).serveConn\ngithub.com/valyala/fasthttp@v1.24.0/workerpool.go:223                    (*workerPool).workerFunc\ngithub.com/valyala/fasthttp@v1.24.0/workerpool.go:195
(*workerPool).getCh.func1\nruntime/asm_amd64.s:1371                                                  goexit"
```

For more information

If you have any questions or comments about this advisory:

- Open an issue at authelia
- Email us at security@authelia.com

Severity

Critical **10.0** / 10

CVSS base metrics

Attack vector	**Network**
Attack complexity	**Low**
Privileges required	**None**
User interaction	**None**
Scope	**Changed**
Confidentiality	**High**
Integrity	**High**
Availability	**High**

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

CVE ID

CVE-2021-32637

Weaknesses

CWE-287