Arbitrary File Deletion via Path Traversal in `removeUnusedAttributeView`

High 88250 published GHSA-vw86-c94w-v3x4 last week

Package

siyuan (Go)

Affected versions

<=3.6.3

Patched versions

v3.6.4

Description

Summary

The endpoint `/api/av/removeUnusedAttributeView` is vulnerable to a **path traversal (CWE-22)** that allows an attacker to delete arbitrary `.json` files on the server.

The issue arises because user-controlled input (`id`) is directly used in filesystem path construction without validation or restriction.

> Access to this endpoint (e.g., via a Reader-role or publish context) is considered a precondition and not part of the vulnerability. The root cause is unsafe path handling.

Steps To Reproduce

1. Ensure the target instance has the publish service enabled (or any valid access to the endpoint).
2. Send the following request:

```
POST /api/av/removeUnusedAttributeView HTTP/1.1
Host: <target>
Content-Type: application/json
```

```
{
  "id": "../../../conf/conf"
}
```

3. Observe that the request is accepted.
4. The server resolves the path outside the intended directory and deletes the target file.

Impact

An attacker can delete arbitrary `.json` files within the workspace directory.

This may lead to:

- Deletion of global configuration files (e.g., `conf/conf.json`)
- Loss of user data and application state
- Corruption of workspace metadata
- Persistent application instability or forced recovery

This represents a **server-side arbitrary file deletion primitive**, which can have severe impact depending on the targeted files.

Technical Details

The vulnerable code constructs file paths as follows:

```
filepath.Join(util.DataDir, "storage", "av", id+".json")
```

Because `id` is not validated, attackers can inject path traversal sequences such as `../` to escape the intended directory.

Example payloads

- `../local` → `data/storage/local.json`
- `../../storage/outline` → `data/storage/outline.json`
- `../../../conf/conf` → `conf/conf.json`

No validation or restriction is applied to:

- input format
- path normalization
- directory boundaries

Root Cause

- Untrusted user input (`id`) is directly used in filesystem path construction
- No input validation or sanitization
- No enforcement that the resolved path stays within the intended directory

Remediation

1. **Validate input strictly**
   - Only allow valid Attribute View IDs
   - Reject any input containing path traversal sequences
2. **Enforce directory boundaries**

```
base := filepath.Join(util.DataDir, "storage", "av")
absPath := filepath.Join(base, id+".json")

if !util.IsSubPath(base, absPath) {
    return error
}
```

3. **Normalize paths before use**
   - Ensure canonical paths cannot escape the base directory
4. **Add additional logical checks**
   - Verify that the target object is valid and allowed to be deleted

Severity

High 8.5 / 10

CVSS v3 base metrics

Attack vector	Network
Attack complexity	Low
Privileges required	Low
User interaction	None

Scope	Changed
Confidentiality	None
Integrity	Low
Availability	High

Learn more about base metrics

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

CVE ID

CVE-2026-40318

Weaknesses

▶ CWE-24

Credits

ch1nhpd Reporter