3 CVE-2022-32207: Unpreserved file permissions

Share:

TIMELINE

nyymi submitted a report to curl. May 17th (6 months ago)

Summary:

Curl fails to preserve file permissions when writing:

- `CURLOPT_COOKIEJAR` database
- `CURLOPT_ALTSVC` database
- `CURLOPT_HSTS` database

Instead the permissions is always reset to 0666 & ~umask if the file is updated.

As a result a file that was before protected against read access by other users becomes other user readable (as long as umask doesn't have bit 2 set).
Out of these files only the `CURLOPT_COOKIEJAR` is likely to contain sensitive information.

In addition curl will replace softlink to the database with locally written database, or if the application is run privileged, specifying `"/dev/null"` as a file name can lead to system overwriting the special file and result in inoperable system.

This is CWE-281: Improper Preservation of Permissions

Steps To Reproduce:

1. `umask 022`
2. `install -m 600 /dev/null cookie.db`
3. `curl -b cookie.db -c cookie.db https://google.com`
4. `ls -l cookie.db`

At least for `CURLOPT_COOKIEJAR` this vulnerability was introduced in https://github.com/curl/curl/commit/b834890a3fa3f525cd8ef4e99554cdb4558d7e1b - this change was introduced to fix a issue https://github.com/curl/curl/issues/4914

Anything else is likely going to end up causing unexpected behaviour, data loss/leaking, or if the user has high enough permissions, damage to the operating system.

Safe cloning of file permissions can only be achieved if the owner / group of the file match the current user (else group permissions might be incorrect). Hence creating a new file and moving it over the old one should IMO only be attempted if the file user and group match that of the previous file.

If a method of creating a new file is still desired, something like this could be attempted to cover the most use cases:

Code 931 Bytes Wrap lines Copy Download

```
1   /* If old file is a regular file attempt creating a new file with same ownership */
2   struct stat st;
3   if (stat(filename, &st) != -1 && S_ISREG(st.st_mode)) {
4     FILE *file;
5     int fd;
6     struct stat nst;
7     fd = open(tempstore, O_CREAT | O_EXCL, 0700);
8     if (fd == -1)
9       goto fail;
10     if (fstat(fd, &nst) == -1 ||
11        nst.st_uid != st.st_uid || nst.st_gid != st.st_gid) {
12       /* newly created file doesn't have same ownership, we can't proceed safely */
13       close(fd);
14       unlink(tempstore);
15       goto fail; // or perhaps try direct write instead?
16      }
17     /* use same mode as old file */
18     if (fchmod(fd, st.st_mode) == -1) {
19       close(fd);
20       unlink(tempstore);
21       goto fail;
22     }
23     file = fdopen(fd, FOPEN_WRITETEXT);
24     if (!file) {
25       close(fd);
26       unlink(tempstore);
27       goto fail;
```

```
31   }
32   else  {
33     /* use direct file write */
34   }
```

Impact

Leak of sensitive information