70 CVE-2021-22901: TLS session caching disaster

Share:

TIMELINE

nyymi submitted a report to curl. Apr 29th (2 years ago)

Summary:

lib/vtls/openssl.c `ossl_connect_step1` sets up the `ossl_new_session_cb` sessionid callback with `SSL_CTX_sess_set_new_cb`, and adds association from `data_idx` and `connectdata_idx` to current `conn` and `data` respectively:

Code 167 Bytes — Wrap lines Copy Download

```
1   SSL_CTX_set_session_cache_mode(backend->ctx,
2         SSL_SESS_CACHE_CLIENT | SSL_SESS_CACHE_NO_INTERNAL);
3   SSL_CTX_sess_set_new_cb(backend->ctx, ossl_new_session_cb);
```

...

Code 118 Bytes — Wrap lines Copy Download

```
1         SSL_set_ex_data(backend->handle, data_idx, data);
2         SSL_set_ex_data(backend->handle, connectdata_idx, conn);
```

Whenever the `ossl_new_session_cb` callback is called the code fetches the `conn` and `data` associated via:

Code 158 Bytes — Wrap lines Copy Download

```
1   conn = (struct connectdata*) SSL_get_ex_data(ssl, connectdata_idx);
2   if(!conn)
3     return 0;
4
5   data = (struct Curl_easy *) SSL_get_ex_data(ssl, data_idx);
```

However, it is possible that the connection is disassociated from these pointers via `Curl_detach_connnection`, and reassociated to a different connection via `Curl_attach_connnection`. Yet, `Curl_detach_connnection` doesn't `SSL_set_ex_data` the `data_idx` / `connectdata_idx` / to NULL, nor does `Curl_attach_connnection` update the pointers with new ones. I am not absolutely certain but this appears to lead to a situation where a stale pointer(s) can exists when the session callback is called.

Steps To Reproduce:

Unfortunately I currently have no easy to way reproduce this issue. I might attempt to do this later.

Notes

This issue is currently lacking information but includes what I believe is the potential root cause of the issue. This information might be wrong or lacking necessary details to make full determination of the validity of this issue at this time.

This issue seems to be occurring somewhat periodically when webkit browser is built with the libcurl backend. Typically this is a rare use case, I know of only Sony Playstation devices that use in larger scale.

Impact

Use after free, with potential for (remote(*)) code execution as `ossl_new_session_cb` calls `Curl_ssl_sessionid_lock(data);` with potentially repurposed memory. Attacker would need to control `data->share` pointer to attacker controller memory. This fake `struct Curl_share` would need to be crafted in a way that `if(share->specifier & (1<<type))` is taken. `share->lockfunc` would then get called by the function, resulting in code execution.

*) caveat here, as it is unknown if external attacker can trigger this situation. It would be difficult, but cannot be completely ruled out.