New issue

Jump to bottom

(Remote DOS attack) 0day buffer overflow vulnerability reveal #629

Closed Icejl opened this issue on Mar 23, 2020 · 6 comments

Icejl commented on Mar 23, 2020 • edited

Hi, Memcached team,

Recently, I revealed a buffer overflow vulnerability which may cause DOS attack. The exploit details can be found as following.

Affect Version

memcached-1.6.0
memcached-1.6.1

Root cause

file location: memcached.c:6156-6187

```
6151            return -1;
6152        }
6153
6154        uint8_t extlen = c->binary_header.request.extlen;
6155        uint16_t keylen = c->binary_header.request.keylen;
6156        if (c->rbytes < keylen + extlen + sizeof(c->binary_header)) {
6157            // Still need more bytes. Let try_read_network() realign the
6158            // read-buffer and fetch more data as necessary.
6159            return 0;
6160        }
6161
6162        if (!resp_start(c)) {
6163            conn_set_state(c, conn_closing);
6164            return -1;
6165        }
6166
6167        c->cmd = c->binary_header.request.opcode;
6168        c->keylen = c->binary_header.request.keylen;
6169        c->opaque = c->binary_header.request.opaque;
6170        /* clear the returned cas value */
6171        c->cas = 0;
6172
6173        c->last_cmd_time = current_time;
6174        // sigh. binprot has no "largest possible extlen" define, and I don't
6175        // want to refactor a ton of code either. Header is only ever used out
6176        // of c->binary_header, but the extlen stuff is used for the latter
6177        // bytes. Just wastes 24 bytes on the stack this way.
6178        char extbuf[sizeof(c->binary_header) + BIN_MAX_EXTLEN];
6179        memcpy(extbuf + sizeof(c->binary_header), c->rcurr + sizeof(c->binary_header), extlen);
6180        c->rbytes -= sizeof(c->binary_header) + extlen + keylen;
6181        c->rcurr += sizeof(c->binary_header) + extlen + keylen;
6182
6183        dispatch_bin_command(c, extbuf);
6184    }
6185
6186    return 1;
6187 }
6188
```

Code Audit

```
6178   char extbuf[sizeof(c->binary_header) + BIN_MAX_EXTLEN];
6179   memcpy(extbuf + sizeof(c->binary_header), c->rcurr + sizeof(c->binary_header), **extlen**);
```

in line 6179, since there is no mechanism to verify the parameter's length, in this case, the length of "**extlen**" when calling memcpy function, It will cause buffer overflow if large value assigned to the **extlen** variable.

POC

```
0x80 0x01 [0x00 0x00] keylen

[0x30] extlen 0x00 0x00 x00
```

for the POC snippet, first, if I assign a large value to the variable **extlen**, on the other hand, in order to bypass the validation of data packet which sent in following code snippet,

```
6156 if (c->rbytes < keylen + extlen + sizeof(c->binary_header))
```

we can construct a very large data packet and send it to the server running **memcached** 1.6.0 or 1.6.1 anonymously. After that, the program will crash because of the issue mentioned above.

Note: Please confirm this issue ASAP. Besides, just letting you know, I am gonna submit this issue to CVE mitre.

Please let me if you have any questions.

Sincerely,
Icejl

1

Icejl changed the title ~~(DOS attack) 0day buffer overflow vulnerability reveal~~ **(Remote DOS attack) 0day buffer overflow vulnerability reveal** on Mar 23, 2020

thesamesam commented on Mar 23, 2020 • edited

In future, it may be wise to message the maintainer privately first and let them confirm the issue and give them the standard period before releasing either way.

I've messaged the maintainer on IRC, waiting for an update.

1 1

Crest commented on Mar 23, 2020

> In future, it may be wise to message the maintainer privately first and let them confirm the issue and give them the standard period before releasing either way.

Will memcached users be afforded the same consideration by attackers?

1

rubyFeedback commented on Mar 23, 2020

I applaude Icejl here because now people who don't want to get DOSed can disable/remove the buggy memcached version (or fix it and recompile; since it's such a small change anyway). Without that information they may otherwise run vulnerable code, so I am 100% in support of Icejl and completely against thesamesam's suggestion to be silent. "Standard period" means unknown silence and no fixes in that time, so it is a misnomer.

thesamesam commented on Mar 23, 2020 • edited

I didn't come up with the idea of responsible disclosure, I was just suggesting. I don't intend to debate on the merits of it - that's for the wider community to decide. Just letting the OP know that it's a thing.

This happens in a lot of places, and I'm not the person to take it out on if you disagree with it.

memcached locked as **off-topic** and limited conversation to collaborators on Mar 23, 2020

dormando commented on Mar 23, 2020 • edited Member

Locking this off-topic discussion.

dormando commented on Mar 23, 2020 Member

1.6.2 released with fix: https://github.com/memcached/memcached/wiki/ReleaseNotes162
specifically: `02c6a2b`

and fwiw, I've been responsive to security reports (or even report them myself) and give credit happily when due for over ten years. Don't waste my good will, please.

dormando closed this as completed on Mar 23, 2020

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
No branches or pull requests

5 participants