

[New issue](#)[Jump to bottom](#)

OpenSSL error 140E0197 with Qt >= 5.12.2 #3679

🔒 Closed an0nfunc opened this issue on Apr 29, 2019 · 42 comments · Fixed by [#4032](#)

Labels priority/P0 - Blocker qt upstream-bug

Milestone 📌 1.3.1

an0nfunc commented on Apr 29, 2019 • edited

After upgrading to Qt 5.12.2 all clients got a disconnect after about 5 minutes of connection time. Reproducible with Qt 5.12.2 and Qt 5.12.3, downgrading to 5.12.1 solves this issue.

Not sure if this is a Qt bug or not, so I'm reporting it here first, however it looks to me like a bug in Qt.

[murmur.log](#)

Mumble/Murmur 1.3.0_rc1
Archlinux

Edit: Only the Qt version of murmur is relevant, client runs latest Qt just fine with murmur on Qt 5.12.1.

metsuke0 commented on Jun 16, 2019

Confirmed this exact issue happens to me and another user using the rc1 client.

reelsense commented on Jul 15, 2019 • edited

Same thing on FreeBSD 11.3-RELEASE with `murmur-1.3.0.rc1_4` via `pkg`.

Clients connect and get disconnected instantly with this log entry:

```
Connection closed: Error while reading: error:140E0197:SSL routines:SSL_shutdown:shutdown while in init
```

Same thing with `murmur-1.3.0.rc1_5` via ports.

  **davidebeatrici** pinned this issue on Jul 15, 2019

davidebeatrici commented on Jul 23, 2019

Member

I failed to reproduce the issue on FreeBSD 12.0 (Qt 5.12.2) and Fedora 30 (Qt 5.12.4).

an0nfunc commented on Jul 23, 2019

Author

Just confirmed it's happening with Qt 5.13.0 and murmur 1.3.0-rc2 as well.

Steps to reproduce:

1. Start murmur
2. Connect with client (version seems to be unrelated)
3. Sit in server for about 5 minutes (can vary depending on murmur uptime)
4. Get disconnected with server-side error message `Connection closed: Error while reading: error:140E0197:SSL routines:SSL_shutdown:shutdown while in init [20]`

an0nfunc commented on Sep 9, 2019 • edited

Author

After some testing it seems like the database used is somehow related. I got no problems after switching the same config from the QMYSQL adapter to sqlite. Same settings otherwise.

Just to confirm, [@metsuke0](#) & [@reelsense](#), what database backend are you using?

EDIT: I'm gonna try and get a SQL log from mariadb for this later today.

reelsense commented on Sep 9, 2019

I'll try to look into this again some time in the next week. I have a feeling it has to do with my large(900MB) sqlite database and murmur 1.3. I moved my murmur server to Ubuntu 18.04 and it runs 1.2.19 PPA.

reelsense commented on Oct 31, 2019

I didn't have time to figure out what dependency was having a problem. So I moved all my mumble servers to Ubuntu 18.04. And since I upgraded them to the new murmur 1.3 release I haven't had a problem.

  **davebeatrici** added bug server priority/P0 - Blocker labels on Mar 19, 2020

  **davebeatrici** added this to the 1.3.1 milestone on Mar 19, 2020

davebeatrici commented on Mar 19, 2020

Member

This is a critical issue, we received a private report this week explaining exactly how to trigger it.

We will probably disclose it through a CVE as soon as it's fixed and 1.3.1 is released.

  **davebeatrici** changed the title ~~Connection drops with Qt 5.12.2 (SSL_shutdown:shutdown while in init)~~ SSL routines:SSL_shutdown:shutdown while in init on Mar 19, 2020

cfstras commented on Mar 22, 2020 • edited

Contributor

We've also noticed this trying to build with a new OpenSSL (using the Dockerfile, with the unreleased Ubuntu focal instead of disco).

Interestingly, only some users randomly get kicked out every few minutes.


Example log:

```
mumble_1      | <W>2020-03-22 21:43:05.034 1 => <34:(-1)> Connection closed: Error during SSL handshake: error:14209102:SSL
routines:tls_early_post_process_client_hello:unsupported protocol [13]
mumble_1      | <W>2020-03-22 21:43:05.050 1 => <33:user1234 (12)> Connection closed: Error while reading: error:140E0197:SSL routines:SSL_shutdown:shutdown while in init [20]
```

bin commented on Mar 25, 2020 • edited



@**davebeatrici** Seeing this issue frequently as well, also with an arch server. Is there any mitigation for this aside from an update, or if not, an ETA on 1.3.1? I haven't seen it on plumbe (third-party android app), but have had pymumble bots with the same issue. It also seems to occur with more users more frequently. The server sees clients as ghosts and disconnects them, but client reports the remote host closed the connection. Also mentions being unable to send UDP packets and switching to TCP mode, and as @AnonFuncsAreAwesome mentioned, disconnects usually every five minutes or so (though has disconnected twice in a row within ~15 seconds before).

Update: tested on an alpine server and had the same result.

 **cfstras** pushed a commit to flipdot/mumble-docker that referenced this issue on Mar 28, 2020

switch images... 

b03d19b

  **cfstras** mentioned this issue on Mar 28, 2020

switch images... flipdot/mumble-docker#13



cfstras commented on Mar 28, 2020 • edited

Contributor

@**bin** as mentioned above, downgrading Qt on the server should help. Alternatively, you can use a static build, such as the one used in this docker image: <https://github.com/PHLAK/docker-mumble>

bin commented on Mar 28, 2020

@**cfstras** Do this for client, server, or both?

cfstras commented on Mar 28, 2020

Contributor

@**bin** on the server. I wouldn't expect any user of my mumble server to use specific versions of anything...

bin commented on Mar 28, 2020

Thanks, so static linking against QT 15.2.1 should be a safe work-around?

cfstras commented on Mar 28, 2020

Contributor

@**bin** you will have to try that for yourself.;

TredwellGit commented on Apr 1, 2020

Contributor

I stumbled upon an astonishingly easy way to trigger this reliably as a denial of service. @**davebeatrici**, do you need help tracking down exactly where this issue is? Reason I ask is because I don't know if you are working on this privately and I don't want to waste time debugging this if you already know how to fix it.

davebeatrici commented on Apr 1, 2020 • edited

Member

Thank you for asking!

We're currently finishing up the new build environment and system, so that all developers can easily build Mumble.

Help is of course appreciated to find the root cause of this issue.

The following function is called right after the warning by OpenSSL:

[mumble/src/murmur/Server.cpp](#)
Lines 1424 to 1475 in b54cf63

```
1424 void Server::connectionClosed(QAbstractSocket::SocketError err, const QString &reason) {
1425     Connection *c = qobject_cast<Connection *>(<sender());
1426     if (! c)
1427         return;
1428     if (c->bDisconnectedEmitted)
1429         return;
1430     c->bDisconnectedEmitted = true;
1431     // ...
1432     // ...
1433     // ...
```

Returning early from the function prevents the client from being disconnected.

It's definitely an issue related to Qt, but I'm not sure whether it's due to a bug or wrong API usage and we're only now seeing the effects due to internal Qt changes.

TredwellGit commented on Apr 2, 2020

Contributor

Found it. In [qt/qtbase@ 93a803a](#) Qt incorrectly calls `SSL_shutdown()` in OpenSSL mid-handshake. I compiled [qt5-base](#) on Arch Linux with that commit reverted and Mumble no longer disconnects clients or shows the error message.

1

bin commented on Apr 2, 2020 • edited

@TredwellGit What condition has caused this to trigger for you on the murmur server?

TredwellGit commented on Apr 2, 2020

Contributor

I don't feel that comfortable sharing how to trigger it because it is a very effective denial of service attack and since @davidebeatrici already stated in [#3679 \(comment\)](#) that he wants to wait for it to be fixed.

davidebeatrici commented on Apr 2, 2020 • edited

Member

Found it. In [qt/qtbase@ 93a803a](#) Qt incorrectly calls `SSL_shutdown()` in OpenSSL mid-handshake. I compiled qt5-base on Arch Linux with that commit reverted and Mumble no longer disconnects clients or shows the error message.

@TredwellGit Good job!

I actually had found that commit too, but didn't investigate further because I mistakenly thought `QsSISocketBackendPrivate::destroySslContext()` was only called after the session is closed.

From https://www.openssl.org/docs/manmaster/man3/SSL_shutdown.html:

`SSL_shutdown()` only closes the write direction. It is not possible to call `SSL_write()` after calling `SSL_shutdown()`. The read direction is closed by the peer.

The behaviour of `SSL_shutdown()` additionally depends on the underlying BIO. If the underlying BIO is **blocking**, `SSL_shutdown()` will only return once the handshake step has been finished or an error occurred.

If the underlying BIO is **non-blocking**, `SSL_shutdown()` will also return when the underlying BIO could not satisfy the needs of `SSL_shutdown()` to continue the handshake. In this case a call to `SSL_get_error()` with the return value of `SSL_shutdown()` will yield `SSL_ERROR_WANT_READ` or `SSL_ERROR_WANT_WRITE`. The calling process then must repeat the call after taking appropriate action to satisfy the needs of `SSL_shutdown()`. The action depends on the underlying BIO. When using a non-blocking socket, nothing is to be done, but `select()` can be used to check for the required condition. When using a buffering BIO, like a BIO pair, data must be written into or retrieved out of the BIO before being able to continue.

After `SSL_shutdown()` returned 0, it is possible to call `SSL_shutdown()` again to wait for the peer's `close_notify` alert. `SSL_shutdown()` will return 1 in that case. However, it is recommended to wait for it using `SSL_read()` instead.

`SSL_shutdown()` can be modified to only set the connection to "shutdown" state but not actually send the `close_notify` alert messages, see [SSL_CTX_set_quiet_shutdown\(3\)](#). When "quiet shutdown" is enabled, `SSL_shutdown()` will always succeed and return 1.

@bin Please send me an email, I will reply with details.

Krzmbzrl commented on Apr 2, 2020

Member

Qt incorrectly calls `SSL_shutdown()` in OpenSSL mid-handshake.

Does this mean this is ultimately a Qt bug? If so has it been reported already? If it has not, we should probably report it in order to get it fixed...

an0nfunc commented on Apr 2, 2020

Author

More interestingly is why only mumble seems to be affected. Is mumble doing something unique with Qt's SSL implementation?

2

bendem commented on Apr 2, 2020

Contributor

Just in case, https://wiki.qt.io/Reporting_Bugs#Bugs_with_security-sensitive_information

davidebeatrici commented on Apr 2, 2020 • edited

Member

I actually had found that commit too, but didn't investigate further because I mistakenly thought `QsSISocketBackendPrivate::destroySslContext()` was only called after the session is closed.

Even then, I wouldn't expect other SSL sessions to be affected.

Possible bug in OpenSSL?

OpenSSL's behavior seems to be fine: when returning early from `Server::connectionClosed()`, preventing the function from effectively disconnecting the user, its session is not affected at all.

`Server::connectionClosed()` is triggered by `Connection::connectionClosed()`:

[mumble/src/murmur/Server.cpp](#)

Line 1288 in b54cf63

```
1288     connect(u, SIGNAL(connectionClosed(QAbstractSocket::SocketError, const QString &)), this, SLOT(connectionClosed(QAbstractSocket::SocketError, const QString &)));
```

`Connection::connectionClosed()` is emitted when `QsSISocket::error()` is:

[mumble/src/Connection.cpp](#)

Line 39 in b54cf63

```
39     connect(qtsSocket, SIGNAL(error(QAbstractSocket::SocketError)), this, SLOT(socketError(QAbstractSocket::SocketError)));
```

[mumble/src/Connection.cpp](#)

Lines 140 to 142 in b54cf63

```
140     void Connection::socketError(QAbstractSocket::SocketError err) {  
141         emit connectionClosed(err, qtsSocket->errorString());  
142     }
```

It seems like Qt is emitting `error()` in the wrong `QsSISocket`.

TredwellGit commented on Apr 2, 2020

Contributor

Does this mean this is ultimately a Qt bug? If so has it been reported already? If it has not, we should probably report it in order to get it fixed...

It appears to be a Qt bug. Calling `SSL_shutdown()` mid-handshake is incorrect and not handling the resulting error is really incorrect. See how NGINX handles this:

https://github.com/nginx/nginx/blob/65ae8b315211988a821bdc32050768f41571ddae/src/event/ngx_event_openssl.c#L2732-L2824

More interestingly is why only mumble seems to be affected. Is mumble doing something unique with Qt's SSL implementation?

What other server programs are you aware of that use Qt's TLS implementation?

Even then, I wouldn't expect other SSL sessions to be affected.

This is what is confusing me.

  davebeatrici mentioned this issue on Apr 4, 2020

[src/murmur/Server.cpp: implement workaround for critical QSsISocket issue #4032](#)

 Merged

  davebeatrici changed the title `SSL routines:SSL_shutdown:shutdown while in init` OpenSSL error 140E0197 with Qt >= 5.12.2 on Apr 4, 2020

davebeatrici commented on Apr 4, 2020

Member

Workaround in [#4032](#).

@TredwellGit Could you open an issue in [Qt Bug Tracker](#), please?

I can do it too, but you deserve the credits since you're the one who pinpointed the exact line causing the bug.

  Krzembrzl added `upstream-bug` `qt` and removed `bug` `server` labels on Apr 4, 2020

 davebeatrici closed this as completed in [#4032](#) on Apr 4, 2020

TredwellGit commented on Apr 5, 2020

Contributor

Do you know why Qt emits `error()` from unrelated `QSsISocket(s)`? Is this an additional bug?

I will open an issue report, but I am really more concerned about this getting fixed than I am getting credit.

davebeatrici commented on Apr 5, 2020

Member

I believe it's an additional bug that was already present and didn't cause any issues until [qt/qtbase@93a803a](#).

davebeatrici commented on Apr 12, 2020

Member

@TredwellGit Did you open an issue in Qt's tracker yet?

TredwellGit commented on Apr 13, 2020

Contributor

<https://bugreports.qt.io/browse/QTBUG-83450>

 1

davidebeatrici commented on Apr 13, 2020

Member

Excellent, the description clearly explains the issue in detail.

🔖 Krzmrbrzl unpinning this issue on Apr 14, 2020

davidebeatrici commented on Apr 15, 2020

Member

<https://codereview.qt-project.org/c/qt/qtbase/+297147>

davidebeatrici commented on Apr 23, 2020

Member

Fixed: [qt/qtbase@ 36a8bdb](#)

Krzmrbrzl commented on Apr 24, 2020

Member

Any indication in which version this will make it?

davidebeatrici commented on Apr 24, 2020

Member

No, but the fix has been merged in the 5.15 , 5.14 and 5.12 branches.

Krzmrbrzl commented on Apr 24, 2020

Member

So I guess we can surround our workaround by an `#if QT_VERSION < QT_VERSION_CHECK(5, 15, 0)` , right?

davidebeatrici commented on Apr 30, 2020

Member

Yes, we can.

However, I would also like to disable the workaround for the next 5.12 and 5.14 versions.

Krzmrbrzl commented on Apr 30, 2020

Member

However, I would also like to disable the workaround for the next 5.12 and 5.14 versions.

I guess as long as we're checking for the minor version number as well, this should be fine :)

davidebeatrici commented on Jun 8, 2020

Member

Qt 5.12.9: [qt/qtbase@ 36a8bdb](#)
Qt 5.14.3: [qt/qtbase@ 8ddffc6](#)

an0nfunc commented on Jun 8, 2020

Author

Glad that this was resolved 🙌

davidebeatrici commented on Jun 9, 2020 • edited

Member

We requested a CVE ID for Mumble, but MITRE created it for Qt because it's effectively an upstream issue: [CVE-2020-13962](#)

Assignees

No one assigned

Labels

priority/P0 - Blocker qt upstream-bug

Projects

None yet

Milestone

1.3.1

Development

Successfully merging a pull request may close this issue.

🔗 [src/murmur/Server.cpp: implement workaround for critical QSocket issue](#)
davidebeatrici/mumble

9 participants

