New issue

Jump to bottom

How to use `new tls.TLSSocket(...)` to establish a secure connection? #43994

Open armanbilge opened this issue on Jul 26 · 1 comment

Labels doc tls

armanbilge commented on Jul 26

Affected URL(s)

https://nodejs.org/docs/latest-v18.x/api/tls.html

Description of the problem

What is the correct, non-deprecated way to use the `new tls.TLSSocket(...)` constructor to establish a secure connection? Context: GHSA-2cpx-6pqp-wf35

According to two unmerged docs PRs, when directly calling `new tls.TLSSocket(...)` it is the user's responsibility to validate peer certificates and identity.

In #10846 it says:

> ⚠ *Warning*: When directly constructing a `tls.TLSSocket` instead of using [`tls.connect()`][] it is the caller's responsibility to:
>
> - manage the lifetime of the the underlying socket, including connecting it;
> - validate the peer certificate and identity, see the [`'secure'`][] event.

> Before using the connection, the user *must* make the following checks or the connection should be considered completely insecure:
>
> 1. Verify that the peer certificate is valid, see [`ssl.verifyError()`][].
> 2. Verify that the peer certificate is for the expected host, see [`tls.checkServerIdentity()`][] and [`tls.TLSSocket.getPeerCertificate()`][].

In #23915 it says:

> It is important to remember, however,
> that it is the caller's responsibility to manage the lifecycle of the provided `net.Socket`, including establishing the connection and validating peer certificates and identity. See the [`'secure'`][] event.

And includes an example:

```
tlsSocket.on('secure', function() {
  const err = this.verifyError() ||
    tls.checkServerIdentity(hostname, this.getPeerCertificate());
  if (err)
    this.destroy(err);
});
```

Both PRs demonstrate how to do this validation, but require use of:

1. The `'secure'` event. In the current Node.js documentation, the only mention of `'secure'` is under the deprecated `tls.SecurePair`, and is itself deprecated. It is also not clear that the `'secure'` event is also emitted on `tls.TLSSocket`.
   https://nodejs.org/docs/latest-v18.x/api/tls.html#event-secure
2. `tlsSocket.ssl.verifyError()`, which does not appear at all in the current documentation. Furthermore, according to **TLSCallbacks => TLSWrap, better TLS inception** #840 (comment) `tlsSocket.ssl` is a "legacy property".

Note that the described validation steps appear to be consistent with internal use

node/lib/_tls_wrap.js
Line 1106 in 5fbf33e

```
1106      socket.on('secure', onServerSocketSecure);
```

node/lib/_tls_wrap.js
Lines 1044 to 1055 in 5fbf33e

```
1044    function onServerSocketSecure() {
1045      if (this._requestCert) {
1046        const verifyError = this._handle.verifyError();
1047        if (verifyError) {
1048          this.authorizationError = verifyError.code;
1049
1050          if (this._rejectUnauthorized)
1051            this.destroy();
1052        } else {
1053          this.authorized = true;
1054        }
1055      }
```

This leaves me with two concerns:

1. The current documentation does not indicate that using `new tls.TLSSocket(...)` by itself does not result in a secure connection.
2. As far as I can tell it is impossible to use `new tls.TLSSocket(...)` to establish a secure connection without relying on APIs that are undocumented, deprecated, and/or legacy.

 **armanbilge** added the doc label on Jul 26

 **VoltrexKeyva** added the tls label on Jul 26

mcollina commented on Jul 26 — Member

@tniessen you might be able to help.

Assignees

No one assigned

Labels

doc tls

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

3 participants