Maintaining OpenSSL

This document describes how to update `deps/openssl/`.

If you need to provide updates across all active release lines you will currently need to generate four PRs as follows:

- a PR for master which is generated following the instructions below for OpenSSL 3.x.x.
- a PR for 16.x following the instructions in the v16.x-staging version of this guide.
- a PR for 14.x following the instructions in the v14.x-staging version of this guide.
- a PR which uses the same commit from the third PR to apply the updates to the openssl source code, with a new commit generated by following steps 2 onwards on the 12.x line. This is necessary because the configuration files have embedded timestamps which lead to merge conflicts if cherry-picked from the second PR.

Use of the quictls/openssl fork

Node.js currently uses the quictls/openssl fork, which closely tracks the main openssl/openssl releases with the addition of APIs to support the QUIC protocol.

Details on the fork, as well as the latest sources, can be found at https://github.com/quictls/openssl.

Branches are used per OpenSSL version (for instance, https://github.com/quictls/openssl/tree/OpenSSL_1_1_1j+quic).

Requirements

- Linux environment.
- `perl` Only Perl version 5 is tested.
- `nasm` (https://www.nasm.us/) Version 2.11 or higher is needed.
- GNU `as` in binutils. Version 2.26 or higher is needed.

0. Check requirements

```
% perl -v

This is perl 5, version 22, subversion 1 (v5.22.1) built for
x86_64-linux-gnu-thread-multi
(with 60 registered patches, see perl -V for more detail)

% as --version
GNU assembler (GNU Binutils for Ubuntu) 2.26.1
Copyright (C) 2015 Free Software Foundation, Inc.
```

```
...
% nasm -v
NASM version 2.11.08
```

1. Obtain and extract new OpenSSL sources

Get a new source from https://github.com/quictls/openssl/tree/OpenSSL_1_1_1j+quic and copy all files into `deps/openssl/openssl`. Then add all files and commit them. (The link above, and the branch, will change with each new OpenSSL release).

OpenSSL 1.1.1

```
% git clone https://github.com/quictls/openssl
% cd openssl
% git checkout OpenSSL_1_1_1j+quic
% cd ../node/deps/openssl
% rm -rf openssl
% cp -R ../../../openssl openssl
% rm -rf openssl/.git* openssl/.travis*
% git add --all openssl
% git commit openssl
```

The commit message can be written as (with the openssl version set to the relevant value):

```
deps: upgrade openssl sources to OpenSSL_1_1_1j

This updates all sources in deps/openssl/openssl by:
    $ git clone https://github.com/quictls/openssl
    $ cd openssl
    $ git checkout OpenSSL_1_1_1j+quic
    $ cd ../node/deps/openssl
    $ rm -rf openssl
    $ cp -R ../openssl openssl
    $ rm -rf openssl/.git* openssl/.travis*
    $ git add --all openssl
    $ git commit openssl
```

OpenSSL 3.x.x

```
% git clone https://github.com/quictls/openssl
% cd openssl
% cd ../node/deps/openssl
% rm -rf openssl
% cp -R ../../../openssl openssl
% rm -rf openssl/.git* openssl/.travis*
```

```
% git add --all openssl
% git commit openssl
```

```
deps: upgrade openssl sources to quictls/openssl-3.0.2

This updates all sources in deps/openssl/openssl by:
    $ git clone git@github.com:quictls/openssl.git
    $ cd openssl
    $ git checkout openssl-3.0.2+quic
    $ cd ../node/deps/openssl
    $ rm -rf openssl
    $ cp -R ../../../openssl openssl
    $ rm -rf openssl/.git* openssl/.travis*
    $ git add --all openssl
    $ git commit openssl
```

2. Execute `make` in `deps/openssl/config` directory

Use `make` to regenerate all platform dependent files in `deps/openssl/config/archs/`:

```
# On non-Linux machines
% make gen-openssl

# On Linux machines
% make -C deps/openssl/config
```

3. Check diffs

Check diffs to ensure updates are right. Even if there are no updates in openssl sources, `buildinf.h` files will be updated because they have timestamp data in them.

```
% git diff -- deps/openssl
```

Note: On Windows, OpenSSL Configure generates a `makefile` that can be used for the `nmake` command. The `make` command in step 2 (above) uses `Makefile_VC-WIN64A` and `Makefile_VC-WIN32` that are manually created. When source files or build options are updated in Windows, it needs to change these two Makefiles by hand. If you are not sure, please ask @shigeki for details.

4. Commit and make test

Update all architecture dependent files. Do not forget to git add or remove files if they are changed before committing:

```
% git add deps/openssl/config/archs
% git add deps/openssl/openssl/include/crypto/bn_conf.h
% git add deps/openssl/openssl/include/crypto/dso_conf.h
```

```
% git add deps/openssl/openssl/include/openssl/opensslconf.h
% git commit
```

The commit message can be written as (with the openssl version set to the relevant value):

OpenSSL 1.1.1

```
deps: update archs files for OpenSSL-1.1.1

After an OpenSSL source update, all the config files need to be
regenerated and committed by:
   $ make -C deps/openssl/config
   $ git add deps/openssl/config/archs
   $ git add deps/openssl/openssl/include/crypto/bn_conf.h
   $ git add deps/openssl/openssl/include/crypto/dso_conf.h
   $ git add deps/openssl/openssl/include/openssl/opensslconf.h
   $ git commit
```

OpenSSL 3.0.x

```
deps: update archs files for quictls/openssl-3.0.0-alpha-16

After an OpenSSL source update, all the config files need to be
regenerated and committed by:
    $ make -C deps/openssl/config
    $ git add deps/openssl/config/archs
    $ git add deps/openssl/openssl
    $ git commit
```

Finally, build Node.js and run the tests.