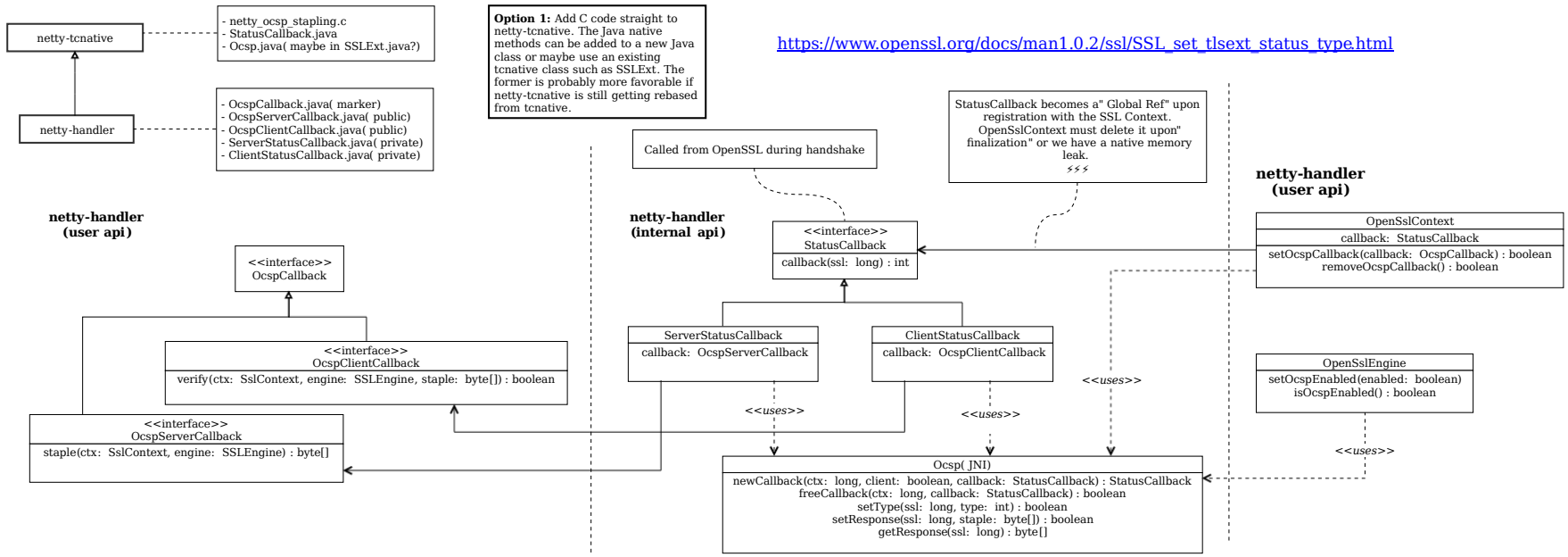
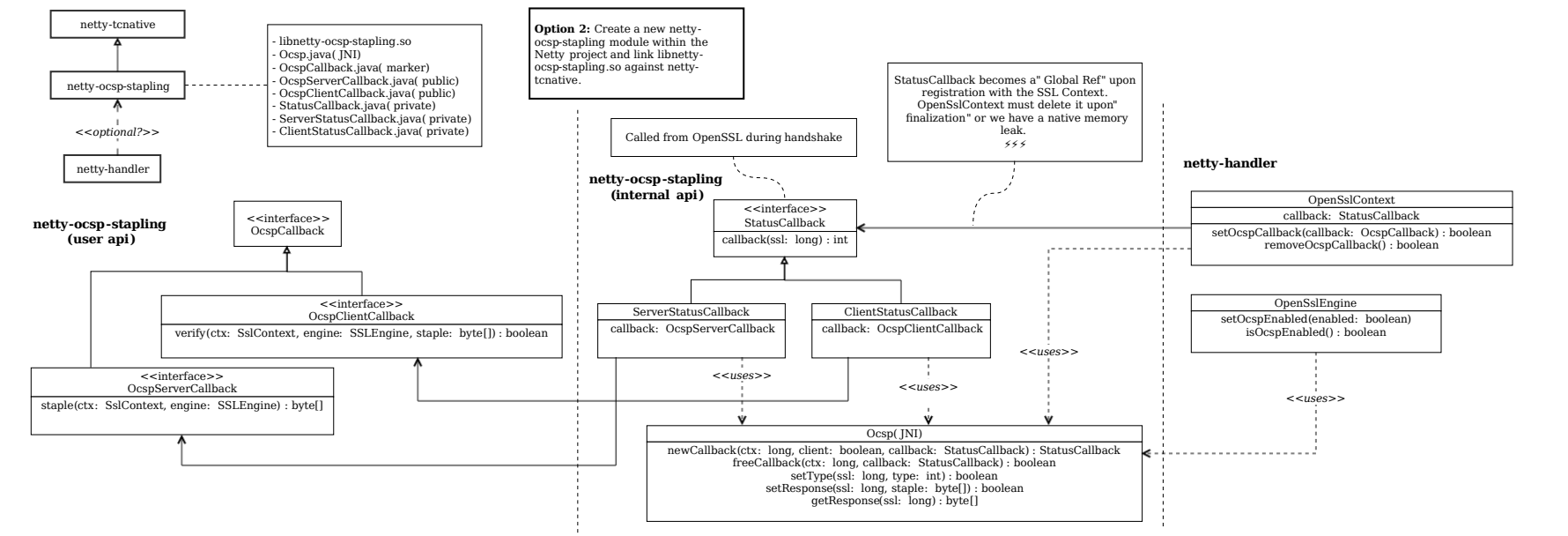

netty-tcnative
- netty_ocsp_stapling.c
- StatusCallback.java
- Ocsp.java(maybe in SSLExt.java?)
netty-handler
- OcspCallback.java(marker)
- OcspServerCallback.java(public)
- OcspClientCallback.java(public)
- ServerStatusCallback.java(private)
- ClientStatusCallback.java(private)
Option 1: Add C code straight to netty-tcnative. The Java native methods can be added to a new Java class or maybe use an existing tcnative class such as SSLExt. The former is probably more favorable if netty-tcnative is still getting rebased from tcnative.
https://www.openssl.org/docs/man1.0.2/ssl/SSL_set_tlsext_status_type.html
Called from OpenSSL during handshake
StatusCallback becomes a" Global Ref" upon registration with the SSL Context. OpenSslContext must delete it upon" finalization" or we have a native memory leak.
⚡⚡⚡
netty-handler (user api)
netty-handler (internal api)
netty-handler (user api)
<<interface>> OcspCallback
<<interface>> StatusCallback
callback(ssl: long) : int
OpenSslContext
callback: StatusCallback
setOcspCallback(callback: OcspCallback) : boolean
removeOcspCallback() : boolean
<<interface>> OcspClientCallback
verify(ctx: SslContext, engine: SSLEngine, staple: byte[]) : boolean
ServerStatusCallback
callback: OcspServerCallback
ClientStatusCallback
callback: OcspClientCallback
OpenSslEngine
setOcspEnabled(enabled: boolean)
isOcspEnabled() : boolean
<<uses>>
<<uses>>
<<uses>>
<<uses>>
<<interface>> OcspServerCallback
staple(ctx: SslContext, engine: SSLEngine) : byte[]
Ocsp(JNI)
newCallback(ctx: long, client: boolean, callback: StatusCallback) : StatusCallback
freeCallback(ctx: long, callback: StatusCallback) : boolean
setType(ssl: long, type: int) : boolean
setResponse(ssl: long, staple: byte[]) : boolean
getResponse(ssl: long) : byte[]


netty-tcnative
netty-ocsp-stapling
<<optional?>>
netty-handler
- libnetty-ocsp-stapling.so
- Ocsp.java(JNI)
- OcspCallback.java(marker)
- OcspServerCallback.java(public)
- OcspClientCallback.java(public)
- StatusCallback.java(private)
- ServerStatusCallback.java(private)
- ClientStatusCallback.java(private)
Option 2: Create a new netty-ocsp-stapling module within the Netty project and link libnetty-ocsp-stapling.so against netty-tcnative.
Called from OpenSSL during handshake
StatusCallback becomes a" Global Ref" upon registration with the SSL Context. OpenSslContext must delete it upon" finalization" or we have a native memory leak.
⚡⚡⚡
netty-ocsp-stapling (internal api)
netty-handler
netty-ocsp-stapling (user api)
<<interface>> OcspCallback
<<interface>> StatusCallback
callback(ssl: long) : int
OpenSslContext
callback: StatusCallback
setOcspCallback(callback: OcspCallback) : boolean
removeOcspCallback() : boolean
<<interface>> OcspClientCallback
verify(ctx: SslContext, engine: SSLEngine, staple: byte[]) : boolean
ServerStatusCallback
callback: OcspServerCallback
ClientStatusCallback
callback: OcspClientCallback
OpenSslEngine
setOcspEnabled(enabled: boolean)
isOcspEnabled() : boolean
<<uses>>
<<uses>>
<<uses>>
<<uses>>
<<interface>> OcspServerCallback
staple(ctx: SslContext, engine: SSLEngine) : byte[]
Ocsp(JNI)
newCallback(ctx: long, client: boolean, callback: StatusCallback) : StatusCallback
freeCallback(ctx: long, callback: StatusCallback) : boolean
setType(ssl: long, type: int) : boolean
setResponse(ssl: long, staple: byte[]) : boolean
getResponse(ssl: long) : byte[]