nghttp2_select_next_protocol

Synopsis

#include <nghttp2/nghttp2.h>

int **nghttp2_select_next_protocol**(unsigned char **out, unsigned char *outlen, const unsigned char *in, unsigned int inlen)

A helper function for dealing with NPN in client side or ALPN in server side. The *in* contains peer's protocol list in preferable order. The format of *in* is length-prefixed and not null-terminated. For example, `h2` and `http/1.1` stored in *in* like this:

```
in[0] = 2
in[1..2] = "h2"
in[3] = 8
in[4..11] = "http/1.1"
inlen = 12
```

The selection algorithm is as follows:

1. If peer's list contains HTTP/2 protocol the library supports, it is selected and returns 1. The following step is not taken.
2. If peer's list contains `http/1.1`, this function selects `http/1.1` and returns 0. The following step is not taken.
3. This function selects nothing and returns -1 (So called non-overlap case). In this case, *out* and *outlen* are left untouched.

Selecting `h2` means that `h2` is written into **out* and its length (which is 2) is assigned to **outlen*.

For ALPN, refer to https://tools.ietf.org/html/rfc7301

See http://technotes.googlecode.com/git/nextprotoneg.html for more details about NPN.

For NPN, to use this method you should do something like:

```
static int select_next_proto_cb(SSL* ssl,
                                unsigned char **out,
                                unsigned char *outlen,
                                const unsigned char *in,
                                unsigned int inlen,
                                void *arg)
{
    int rv;
    rv = nghttp2_select_next_protocol(out, outlen, in, inlen);
    if (rv == -1) {
        return SSL_TLSEXT_ERR_NOACK;
    }
    if (rv == 1) {
        ((MyType*)arg)->http2_selected = 1;
    }
    return SSL_TLSEXT_ERR_OK;
}
...
SSL_CTX_set_next_proto_select_cb(ssl_ctx, select_next_proto_cb, my_obj);
```

Previous Next