Application Layer Protocol Negotiation

TLS extension for application layer protocol negotiation within the TLS handshake

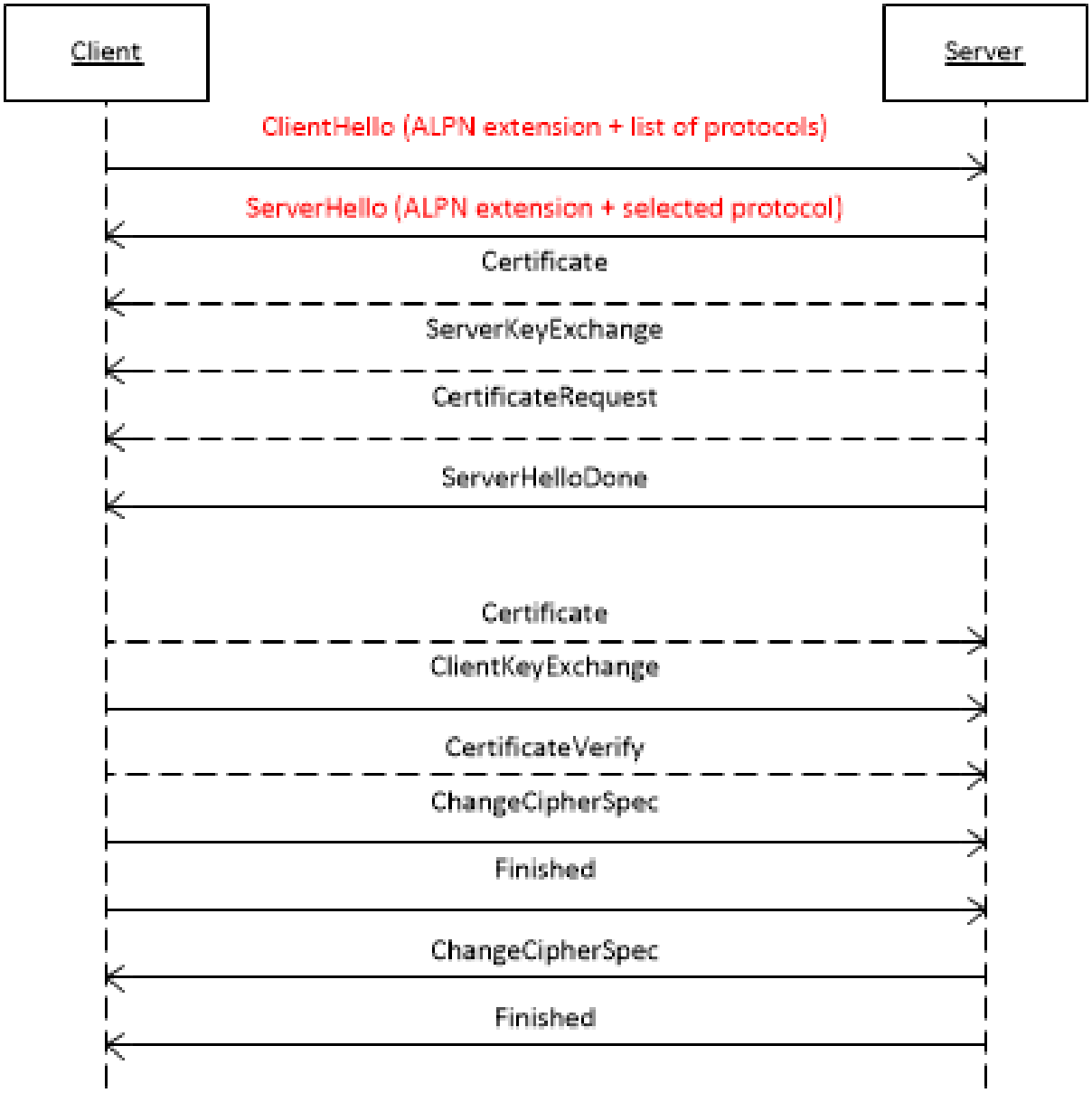
Background and Design Goals

HTTPBis WG requested TLS support for negotiating application layer protocols such as HTTP 1.1 and HTTP 2.0.

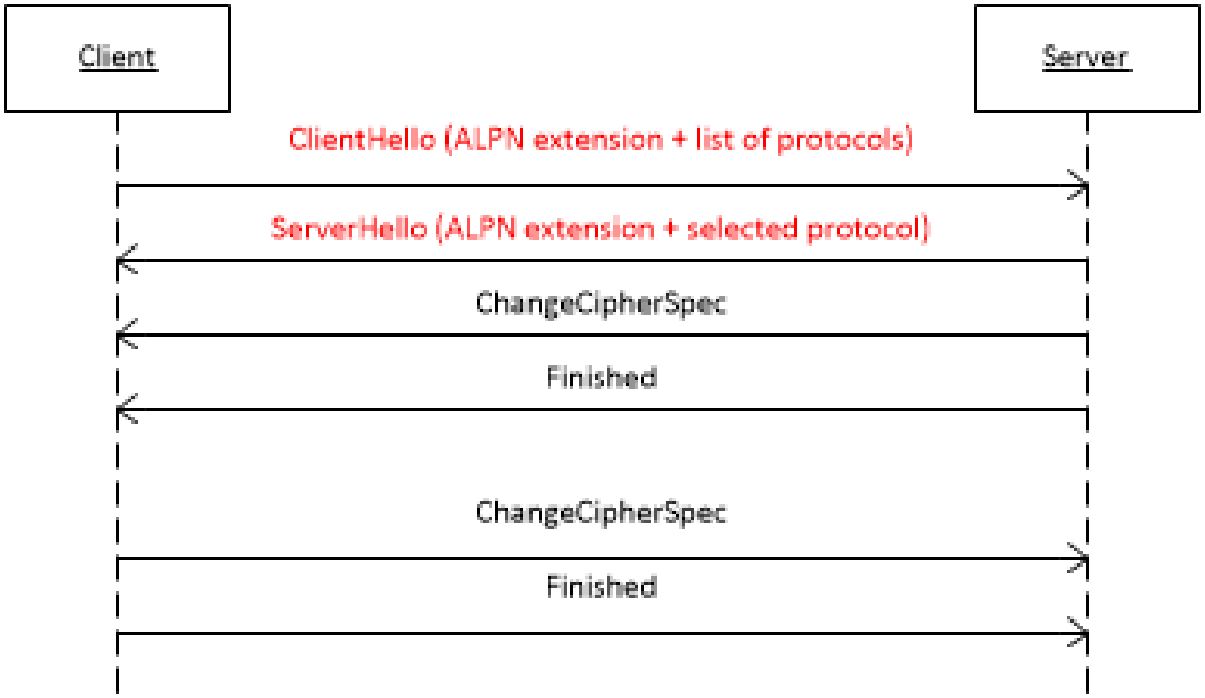
Design goals:

- Negotiate application layer protocol for the connection.
- Minimize connection latency.
- Align with existing TLS extensions.

Full TLS Handshake with ALPN



Abbreviated TLS Handshake with ALPN



ALPN Extension Structure

- The "extension_data" field of the ALPN extension SHALL contain a "ProtocolNameList" value.

```
opaque ProtocolName<1..2^8-1>;
struct {
        ProtocolName protocol_name_list<2..2^16-1>
} ProtocolNameList;
```

- When sent with the ClientHello message, "ProtocolNameList" contains the list of protocols advertised by the client, in descending order of preference.
- When sent with the ServerHello message, "ProtocolNameList" MUST contain exactly one "ProtocolName“ representing the selected protocol.

Protocol IDs and Protocol Selection

- Protocols IDs are IANA registered, opaque, non-empty byte strings.
- Initial registrations have been requested for HTTP/1.1, SPDY/1, SPDY/2, SPDY/3.
- If the server supports no protocols that the client advertises, the server SHALL respond with a fatal "no_application_protocol" alert.

ALPN Design Considerations

- Protocol selection on the server allows certificate to be chosen based on the negotiated protocol.
- The negotiated protocol is known after the first network roundtrip.
- The "extension_data" field of the ALPN extension allows re-use of the existing parsers.
- TLS renegotiation can be used to negotiate an application protocol with confidentiality.

Changes Since IETF87

- Experimental namespace removed per best current practice RFC 6648.
- HTTP/2 protocol ID removed from the initial registrations with the intent that the HTTPbis WG request the appropriate protocol ID(s).
- More specific protocol registry information in the IANA section.
- TLS working group last call for the ALPN draft has completed.

Available Implementations & Tools

- ALPN is implemented in several HTTP/2 prototypes, including Katana, Mozilla, Chromium, iij-http2, GFE.
- ALPN patch for OpenSSL contributed by Google.
- ALPN support for Wireshark network analyzer contributed by Akamai.

ALPN Deployment

- *.google.com servers have ALPN enabled.
- Google Chrome and IE11 support application protocol negotiation via ALPN.
- F5/BIG-IP FW versions older than 10.2.4 cannot handle ClientHello messages longer than 255 bytes. This is a general issue e.g. when adding cipher suites, extensions, or using SNI with a long server name. The use of ALPN extension can also expose this bug.
- We’re reaching out to sites using obsolete F5/BIG-IP firmware. If you run one of these sites, please upgrade!

Links and Contact Information

- ALPN Draft: http://datatracker.ietf.org/doc/draft-ietf-tls-applayerprotoneg
- OpenSSL implementation of ALPN by Google: http://git.openssl.org/gitweb/?p=openssl.git;a=commit;h=6f017a8f9db3a79f3a3406cf8d493ccd346db691

- Stephan Friedl sfriedl@cisco.com
- Andrei Popov andreipo@microsoft.com
- Adam Langley agl@google.com
- Emile Stephan emile.stephan@orange.com