Application Layer Protocol Negotiation

A 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.

What Has Changed

- Now a WG item, current version is:
  - draft-ietf-tls-applayerprotoneg-01
  - posted April 25th
- Code point assigned
  - IANA registration: application_layer_protocol_negotiation(16)
- Changes since last meeting:
  - Revised Introduction
  - Addition of extension type value
  - Request for IANA to create a registry for “Application Layer Protocol Negotiation (ALPN) Protocol IDs” under existing TLS heading
  - Addition of HTTP/2 in the references section
  - Removal of paragraph on hash calculations
  - Clean up of some references and reference formats
  - Addition of Adam Langley from Google as a co-author
  - Addition of Emile Stephan from France Telecom – Orange as a co-author

Implementation Status

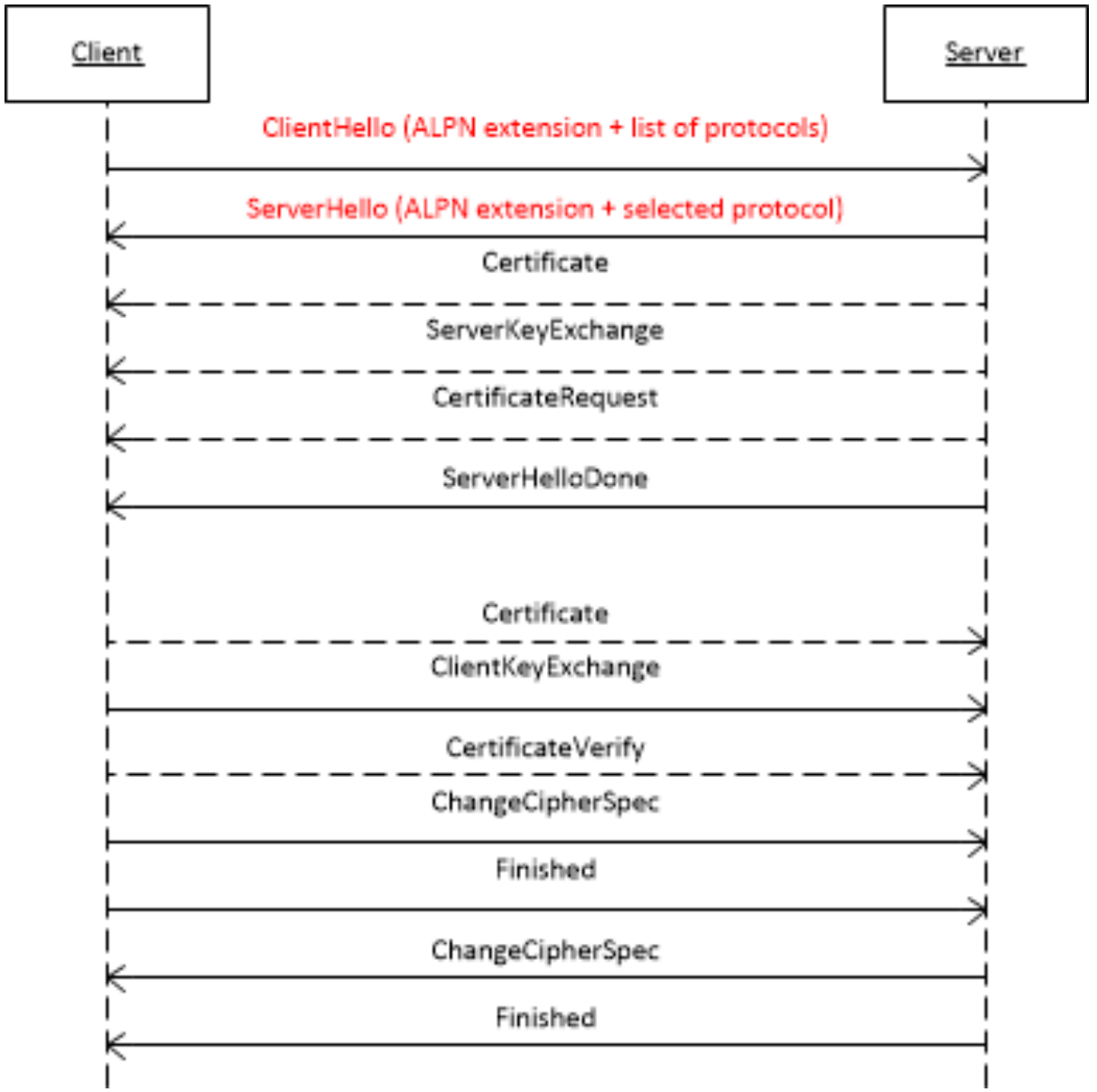
- *.google.com servers support ALPN
- Win 8.1 MP build supports ALPN
- ALPN patch to OpenSSL is in code review

Discussion Status

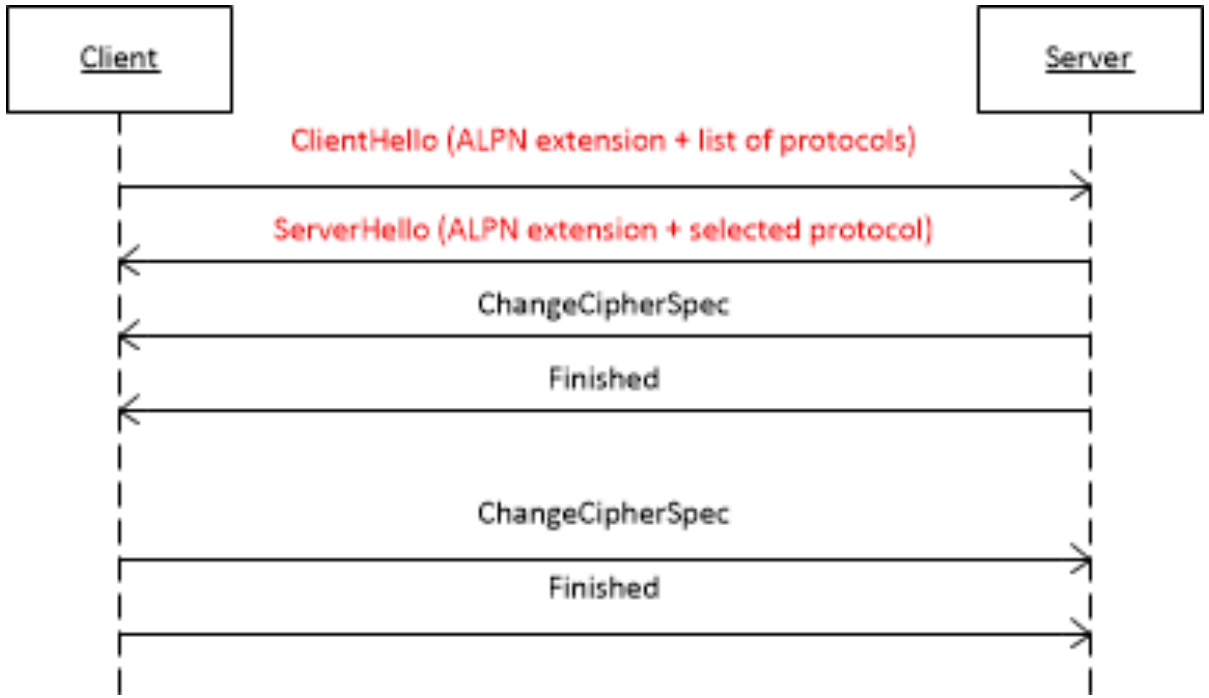
- Some discussion on the mailer, mainly just after the publication of the -00 and -01 drafts
- Believe any open questions have been answered satisfactorily
  - Lingering concerns of no standardized approach to encrypted extensions
    - TLS 1.3 work
  - Request for private application protocol namespace
    - Experimental namespace spec'd, private unnecessary?
  - Request for explicit 'hide' protocol
    - Seems only to complicate protocol and add little
- At present, no further edits planned for draft

Backup

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 are named by IANA registered, opaque, non-empty byte strings.
- A namespace for experimental protocols, which are not registered by IANA, starting with: 0x65, 0x78, 0x70 ("exp").
- 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.

Available Implementations

- MS Open Tech has contributed an open-source reference implementation of ALPN.
- Available as OpenSSL, Apache and mod_spdy patches:

http://html5labs.interopbridges.com/prototypes/alpn/alpn/info

Links and Contact Information

- ALPN Draft: http://datatracker.ietf.org/doc/draft-friedl-tls-applayerprotoneg
- OpenSSL/Apache implementation of ALPN by MS Open Tech: http://html5labs.interopbridges.com/prototypes/alpn/alpn/info

- Stephan Friedl sfriedl@cisco.com
- Andrei Popov andreipo@microsoft.com