Extension for protecting (D)TLS handshakes against Denial of Service

draft-tiloca-tls-dos-handshake-01

Marco Tiloca, RISE SICS
Ludwig Seitz, RISE SICS
Maarten Hoeve, ENCS
Olaf Bergmann, Universitaet Bremen TZI

IETF 100, TLS WG, Singapore, November 16th, 2017

Motivation

- Servers are vulnerable to Denial of Service against (D)TLS handshake
  - Attack: repeatedly send ClientHello messages to victim servers
  - Induce computation, handshake performance, and holding state open
- Cookie exchange
  - Oriented to non on-path adversaries, complicates the attack performance
- Servers still exposed to on-path adversaries
  - Minimally man-on-the-side (can read & inject; echo Cookies, IP spoofing)
  - Maximally full active (can also stop traffic, hold state open at later stages)
- Attack impact
  - Depends on protocol version and used key establishment mode
  - Especially severe on resource-constrained DTLS servers in LLNs

Goal and approach

- Counteract the attack also when mounted by on-path adversaries

- New ClientHello extension “dos_protection”
  - Intended for (D)TLS 1.2 and (D)TLS 1.3
  - Including a Handshake Token opaque to the client
  - The Handshake Token includes a Nonce and a MAC over the Nonce

- A Trust Anchor (TA)
  - In a trust relation with the server
  - Provides the client with the Handshake Token

- The server
  - Checks that the extension is fresh and the MAC is valid
  - Continues with the handshake only in case of positive checks

Protocol overview

- The server is associated with one TA only

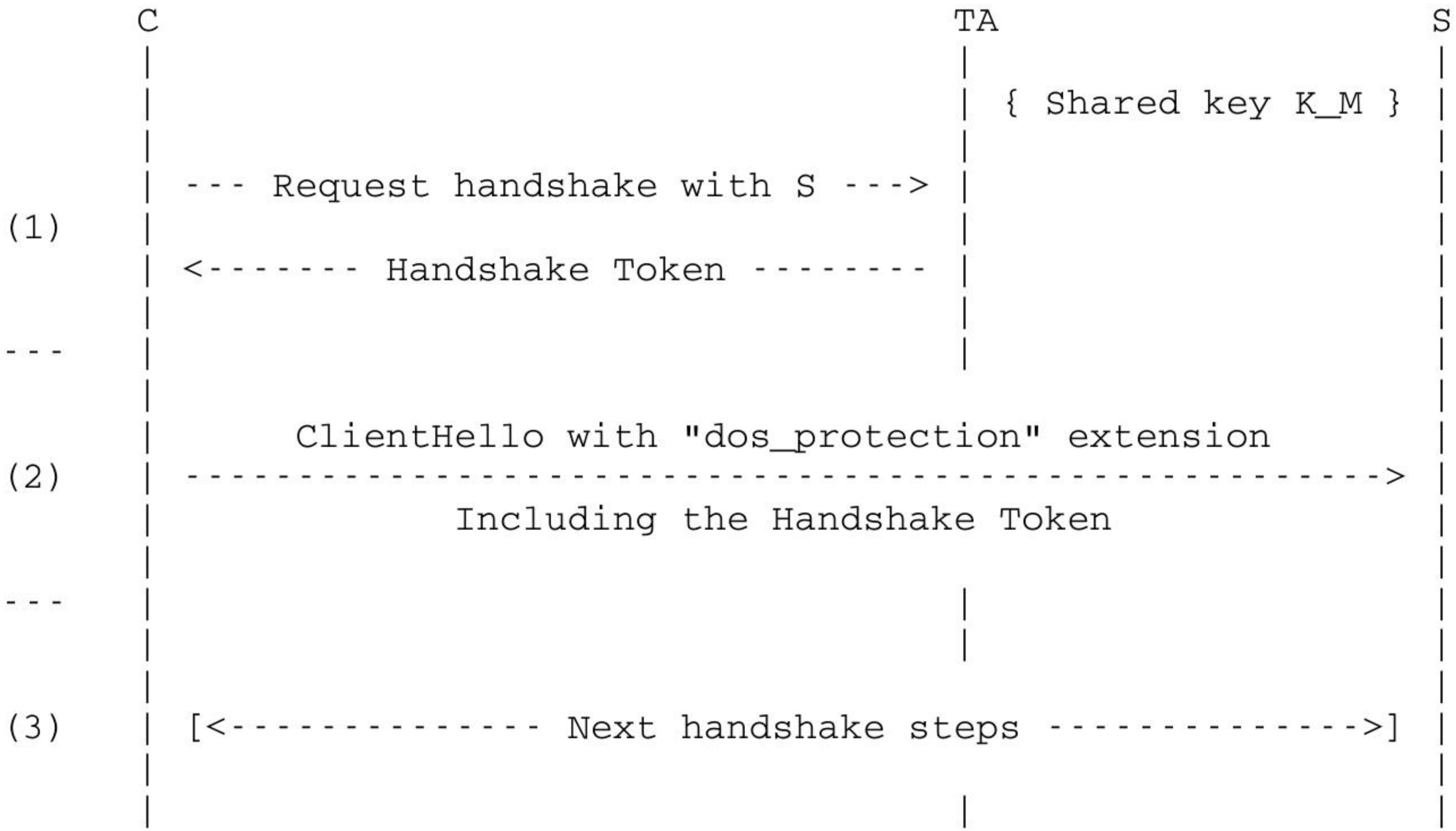
- The server and the TA share a long-term key K_M

- The TA has a pairwise counter z_S per server
  - Initialized to 0 upon the server's registration at the TA
  - Used to build the nonce for the Handshake Token

- The TA verifies the client to be authorized
  - Authorization enforced on the TA or through further trusted parties

- Communications with the TA must be secured
  - Specific means are out of scope

Protocol overview



```
        C                                        TA                       S
        |                                         |                       |
        |                                         |  { Shared key K_M }   |
        |                                         |                       |
        |  --- Request handshake with S --->      |                       |
(1)     |                                         |                       |
        |  <------- Handshake Token --------      |                       |
        |                                         |                       |
---     |                                         |                       |
        |                                                                 |
        |       ClientHello with "dos_protection" extension               |
(2)     |  ------------------------------------------------------------>  |
        |              Including the Handshake Token                      |
        |                                                                 |
---     |                                         |                       |
        |                                         |                       |
        |                                                                 |
(3)     |  [<-------------- Next handshake steps --------------->]        |
        |                                                                 |
        |                                         |                       |
```

Client to TA

- The client contacts the TA
  - Ask to start a new (D)TLS session with the server S

- The TA
  - Uses the counter z_S as token_nonce
  - Computes a MAC as HMAC(K_M, H(token_nonce))
  - Builds the Handshake Token as {token_nonce, MAC}
  - Provides the Handshake Token to the client
  - Increments the counter z_S

- The Handshake Token is opaque to the client
  - The specific semantic is only between the server and the TA

Client to Server

- The client
  - Prepares a “dos_protection” extension including the Handshake Token
  - Includes the extension in the ClientHello message
  - Finalizes the ClientHello and starts the handshake with the server

- The server
  - Checks that the extension is fresh, relying on token_nonce
  - Recomputes the MAC for comparison with the received one
  - In case of negative match, the server aborts the handshake

Additional points (1/2)

- Session resumption
  - This extension is not strictly needed for resumption
  - The server uses the existing association to assert client’s validity
  - Anti-replay checks can rely on the Client Hello Recording mechanism

- Based on Section 7.4.1.4 of RFC5246
  - Clients asking for resumption SHOULD use the same extensions
  - The server would not process the extensions unless relevant

- The TA can provide also Resumption Tokens to the client
  - Used for ClientHello messages sent for session resumption
  - The server does not perform a replay check based on such tokens

Additional points (2/2)

- Replay-check based on the token_nonce
  - A method relying on a sliding window is described in Section 7
  - The window size trades detection accuracy with memory overhead
- Upon a wrap-around of counter z_S
  - Avoid reusing {K_M, Nonce} pairs on the TA
  - The TA MUST revoke K_M and provide the server with a new one
- Rate limit to nonce releases
  - Prevent a client from quickly consuming a server’s nonce space
  - Preserve the TA’s capability to serve other clients

Related document in ACE

- Framework for authentication and authorization in the IoT
  - Based on building blocks including OAuth 2.0 and CoAP
  - Actors involved are Authorization Server, Client, and Resource Server
  - Profiles define the use of concrete transport and security protocols

- DTLS profile of ACE (*)
  - Client and Server establish a DTLS channel
  - Vulnerability to DoS against DTLS handshake is acknowledged
  - Reference to this approach as possible counteraction
  - The ACE Authorization Server acts also as Trust Anchor

(*) *draft-ietf-ace-dtls-authorize-02*

Status and next steps

- Major changes from version -00
  - Same overall approach, with greatly simplified design
  - Improved threat model and security considerations
  - Updates mostly based on a review from Eric Rescorla

- Further comments and feedback are welcome!

- Implementation for DTLS 1.2 in Californium/Scandium
  - Proof-of-concept existing and aligned with version -00
  - To be aligned with current design in version -01

Thank you!

Comments/questions?

https://gitlab.com/crimson84/draft-tiloca-tls-dos-handshake/