Robust (D)TLS-based secure communications

Marco Tiloca[1] , Hamid Rahmouni[2] and Maarten Hooeve[2]
[1]RISE SICS, [2]ENCS
Contact: <marco.tiloca@ri.se>

Abstract

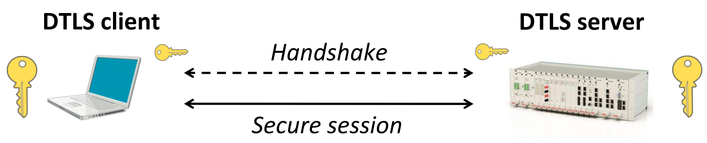
The TLS and DTLS protocols are a valid option to secure communications end-to-end between nodes in a Smart Grid system, such as SCADA units and RTUs in secondary substations. A (D)TLS client and server establish a secure channel through a specific message exchange, namely handshake. However, the handshake itself is vulnerable to a Denial of Service attack, which can induce the server to largely consume resources and start several invalid sessions, hence becoming less responsive or even practically unavailable to serve legitimate clients. The security solution developed in SEGRID allows (D)TLS servers to identify invalid handshakes and promptly abort their execution at the very first step, without breaking the current standards. This considerably limits the attack impact on server units, so practically preserving their availability and the security of the overall Smart Grid system.

Background

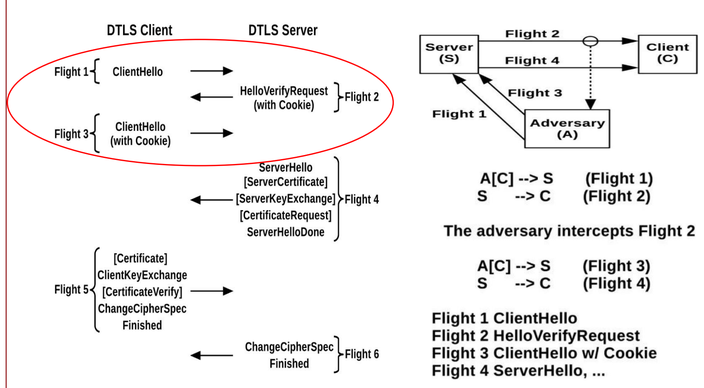
Secure communication among smart grid units
- Data hubs, SCADA units, RTUs in secondary substations, Meter Data Concentrators, …
- Assure authentication, integrity, anti-replay and confidentiality, typically at the transport layer
- TLS and DTLS protocols are available options

Datagram Transport Layer Security (DTLS)
- Essentially, TLS services over UDP
- Prevent eavesdropping and message forgery

Secure communication
- Session establishment as handshake protocol
- Message exchange as record protocol



The handshake is vulnerable to Denial of Service!
- The *ClientHello* message is not authenticated
- An adversary can flood the server for processing *ClientHello* messages and open invalid sessions
- The optional Cookie exchange is not enough
- Server resources and availability greatly impaired

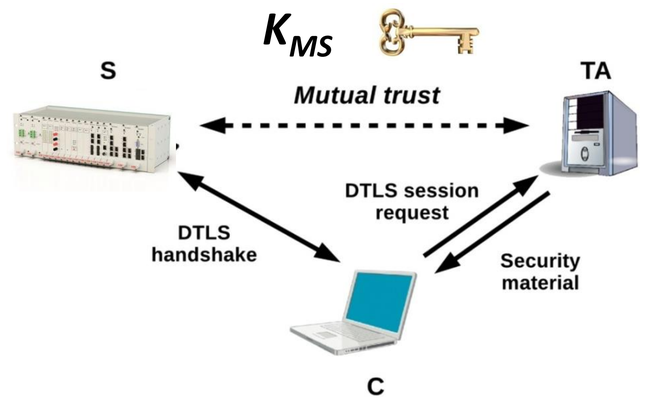


Solution overview

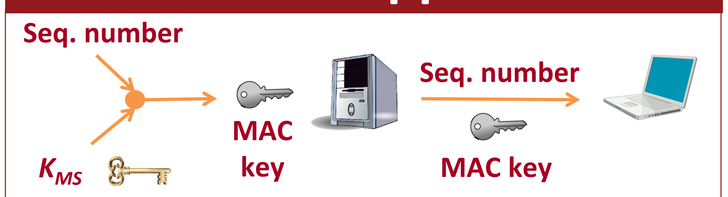
The DTLS server (S):
- Shares a long-term key with a Trust Anchor (TA)
- Locally stores only that key (K_{MS})

Before starting the handshake, the DTLS client (C):
- Contacts the Trust Anchor
- Receives an additional support key

During the handshake:
- The client authenticates the *ClientHello* message
- The server verifies that the *ClientHello* is valid



Solution approach



Step 1 (Client)
- Get a MAC key and a seq. number from the TA
- The MAC key is derived from the shared key K_{MS}

Step 2 (Client)
- Seq. number included in the *ClientHello*
- MAC key used to authenticate the *ClientHello*
- Different techniques are possible (e.g. HMAC)

Step 3 (Client)
- Resulting MAC included in the *ClientHello*
- Use a standard-compliant ClientHello extension

Step 4 (Server)
- Receive the *ClientHello* message
- Retrieve the conveyed seq. number

Step 5 (Server)
- Consider the seq. number and K_{MS}
- Derive the correct MAC key

Step 6 (Server)
- Use the MAC key to verify the *ClientHello*
- Computed MAC compared with the received one

The Cookie exchange is not necessary any longer!

Demo

SEGRID case study
- Communication between RTUs
- First java implementation by RISE SICS
- Porting adaptation and testing by ENCS

Up and running on ENCS equipment
- Enhanced DTLS handshake
- Interaction with the Trust Anchor
- Proof of correct session establishment
- Compared against the original handshake
- Counteraction of a real attack instance

```
Message reception starts at : 971734660157
18 FINEST [DTLSConnector]:  => find handshaker for key /192.168.1.16:60622 - (ch.ethz.inf.vs.scandi
18 FINEST [DTLSConnector]:  => contentType: Handshake (22) - (ch.ethz.inf.vs.scandium.DTLSConnector
18 FINEST [DTLSConnector]:  => handshaker: null - (ch.ethz.inf.vs.scandium.DTLSConnector.java:455)
18 INFO [DTLSConnector]: Created new session as server with peer: /192.168.1.16:60622 - (ch.ethz.in
18 FINEST [DTLSConnector]: Stored handshaker: ch.ethz.inf.vs.scandium.dtls.ServerHandshaker@f97ece
017-01-04 04:26:12)
Bad short MAC in ClientHello message
ShortMAC checked in : 983793 ns
Authenticated ClientHello message processing completed at : 971745256623
Authenticated ClientHello message processed in: 1440716 ns
18 SEVERE [DTLSConnector]: Handshake Exception (/192.168.1.16:60622): Bad short MAC in ClientHello
12)
18 SEVERE [DTLSConnector]: Datagram which lead to exception (/192.168.1.16:60622): 16 FE FD 00 00
C DC 83 14 E9 5E AA CD 5F 00 00 00 04 C0 A8 C0 AC 01 00 00 2B 00 0A 00 08 00 06 00 17 00 18 00 19 C
682) receiveNext() in thread "DTLS-Receiver0.0.0.0/0.0.0.0:4003 at (2017-01-04 04:26:12)
ch.ethz.inf.vs.scandium.dtls.HandshakeException: Bad short MAC in ClientHello message
```

Conclusions

- The Denial of Service attack is neutralized
- The server does not start invalid DTLS sessions
- The server preserves resources and availability
- The Cookie exchange is not required any longer
- Same approach reusable in the TLS protocol

References

M. Tiloca, C. Gehrmann and L. Seitz
On Improving Resistance to Denial of Service and Key Provisioning Scalability of the DTLS Handshake
International Journal of information Security, Vol. 16, issue 2, pp. 173-193, Springer, 2017

M. Tiloca, C. Gehrmann and L. Seitz
Robust and Scalable DTLS Session Establishment
ERCIM News No. 106, Special Theme: Cybersecurity, pp. 31-32, ERCIM, 2016

M. Tiloca, L. Seitz and M. Hoeve
Extension for protecting (D)TLS handshakes against Denial of Service, draft-tiloca-tls-dos-handshake-00
(work in progress), IETF, 2017

Figure 1. Label in 24pt Calibri.

SEGRID Security for smart Electricity GRIDs

www.segrid.eu info@segrid.eu

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 607109