End-to-End Transport Security in the IP-based Internet of Things

Martina Brachmann*, Sye Loong Keoh†, Oscar Garcia Morchon† and Sandeep S. Kumar†

*Computer Networks and Communication Systems Group
Brandenburg University of Technology Cottbus
Walther-Pauer-Str. 2, 03046 Cottbus, Germany
martina.brachmann@tu-cottbus.de

† Department of Lighting Control Systems, Philips Research Europe
High Tech Campus 34, 5656AE, Eindhoven, The Netherlands
{sye.loong.keoh, oscar.garcia, sandeep.kumar}@philips.com

***Abstract*—The IP-based Internet of Things refers to the interconnection of smart devices in a Low-power and Lossy Network (LLN) with the Internet by means of protocols such as 6LoWPAN or CoAP. The mechanisms to protect the LLN from attacks from the Internet and provisioning of an end-to-end (E2E) secure connection are key requirements for functionalities ranging from network access to software updates. Interconnecting such resource constrained devices with high-performance machines requires new security mechanisms that cannot be covered by already known solutions. This paper describes attacks at transport layer against the LLN launched from the Internet. It also introduces approaches to ensure E2E security between two devices located in homogeneous networks using either HTTP/TLS or CoAP/DTLS by proposing a mapping between TLS and DTLS.**

***Index Terms*—Internet of Things, End-to-End security, DTLS, CoAP, Protocol translation.**

I. Introduction

The Internet of Things (IoT) denotes the interconnection of highly heterogeneous networked entities such as sensors, actuators, smart phones, etc. The introduction of IPv6 and web services as fundamental building blocks for IoT applications has created a homogeneous protocol ecosystem, allowing simple integration of IoT devices in a Low-power and Lossy Network (LLN) with Internet hosts. This greatly simplifies the deployment of the envisioned scenarios, ranging from building automation, remote health monitoring, body area networks to personal area networks. Heterogeneous smart objects in an LLN might interact autonomously with each other, controlled by a service portal in the Internet (henceforth called *back-end service*) or a smart phone in the vicinity. The Internet Engineering Task Force (IETF) working group Constrained RESTful Environment (CoRE) aims at providing a framework for resource-oriented applications intended to run on constrained IP networks (e.g., 6LoWPANs). A lightweight alternative of the HTTP protocol, the Constrained Application Protocol (CoAP) that runs over UDP has been defined to enable efficient application-level communication for IoT devices [1].

Security is key for the above application areas and it is crucial that the basic security services such as confidentiality, authentication, and freshness of secret keys between two communicating entities are provided. Information exchanged in the network must be protected end-to-end (E2E). To deal with these security requirements, CoAP offers Datagram Transport Layer Security (DTLS) [2] and when DTLS *NoSec* mode is selected, the CoAP communication could be protected using IPSec [3] at the network layer in an LLN. However, Modagugu and Rescorla [4] have indicated that the use of IPSec may not be suitable in some scenarios, e.g., in CoAP that requires lightweight security. Due to the asymmetry of the system, there are other security considerations around E2E security such as protecting the constrained network from flooding and replay attacks since devices in the LLN have significantly less computational resources and memory when compared to Internet devices.

The paper is organized as follows: In Section II, we present the architectures and assumptions that we consider relevant. Based on this architecture Section III introduces related work on a similar problem. Section IV describes possible attacks and general conclusion of how to prevent them, while Section V discusses possible approaches to mitigate the security issues, with the focus on the mapping between TLS and DTLS to enable secure E2E communication between nodes in a LLN with a back-end service that does not support CoAP. Finally, we conclude the paper with future work.

II. Architectures and Assumptions

The considered architectures are illustrated in Figure 1. It consists of a back-end service in the Internet, a 6LoWPAN Border Router (6LBR) and a group of nodes running CoAP, located in a 6LoWPAN. The 6LBR interconnects the Internet with the LLN, thus allowing for the access to the CoAP/6LoWPAN devices from anywhere in the Internet. This implies that the 6LBR is an authenticated part of the constrained network, achieved by prior bootstrapping mechanism in the LLN. Considering only one 6LBR, the

978-1-4673-1544-9/12/$31.00 ©2012 IEEE

architecture refers to the *Simple 6LoWPAN* as described in [5]. In this architecture, it is assumed that the back-end is a CoAP/HTTP client, while the node in the LLN is implemented as a CoAP server that provides resources and services to be accessed and managed remotely. It is further assumed that the nodes in the LLN are battery operated, whereas the 6LBR is equipped with a power supply. This architecture can be seen as a typical deployment for buildings to facilitate building automation and control.

(a) Back-end is using CoAP to communicate with node

(b) Back-end is using HTTP to communicate with the node

Figure 1. Considered architectures and protocols

As illustrated in Figure 1a, when the two communicating endpoints understand CoAP, E2E security can be provided using DTLS [2]. In this scenario, we consider DTLS with pre-shared keys (PSKs) is used since end-to-end handshake based on PSK requires less computation and resources, hence more suitable for constrained devices when compared with DTLS with public-key (DTLS *RawPublicKey* mode and *Certificate* mode). Since CoAP is not widely supported and deployed in the traditional Internet, proxies or protocol converters are typically necessary to provide translation service between HTTP and CoAP. This is also applicable to the underlying TLS and DTLS protocols. The need for such a translation is mainly because CoAP over TLS is not considered. The reason being TLS needs a reliable connection protocol like TCP, which however incurs a lot of message overhead in a LLN. Using UDP instead of TCP to transport TLS packets however lacks message re-ordering support and retransmission of lost packets. This would lead to catastrophic failure of TLS handshake. Hence, having a back-end system not supporting CoAP (see Figure 1b) requires a mapping of HTTP and CoAP as well as TLS and DTLS. It is thus important to also ensure that a device on the Internet can use HTTP to access resources from a CoAP device directly in a secure manner. In this scenario, the protocol conversion is done by a HTTP/CoAP proxy that is present between them. Subsection V-B.

III. Related Work

A similar architecture as described in Section II was already studied in the context of the Wireless Application Protocol 1.x [6]. To communicate with the Internet, a mobile phone was equipped with the Wireless Application Protocol (WAP), a lightweight version of HTTP. The notion of Wireless TLS (WTLS) was used to secure the WAP between the mobile phone and the Internet. With WTLS, a secure channel is established between the phone and the gateway, serviced by the mobile network operator (MNO) of the mobile phone user. On behalf of the phone, the gateway creates another channel using TLS between itself and the server in the Internet. The phone then can send its WAP messages through the secure channel to the gateway. After decrypting the message, translating it from WAP to HTTP, and encrypting it with the session key of the TLS connection with the server in the Internet, the gateway transfers the message to the server.

There are two security issues with regards to the WAP architecture [7], [8]. The first one concerns the two independent connections that break the end-to-end connectivity between the client (mobile phone) and the server in the Internet. In this case, the communication is interrupted and intercepted by the gateway. The client has no control over the security parameters and cipher suites chosen to protect the communication between the gateway and the server in the Internet. The secret key derived for this connection can be weaker than the client would accept, or it might be possible that this connection is completely unsecured. The second issue is about the translation of WAP and HTTP. When using online banking and exchanging highly confidential data such as credit card numbers or PINs, they would appear in clear text to the gateway during translation of the protocols and therefore readable by the MNO.

Integrated TLS (ITLS) [9] was introduced to remove the security issues occurring in the WAP architecture. The main idea is to prevent the MNO from reading the content of the messages. For this, the WTLS handshake has been modified so that the Client negotiates two keys, one with the gateway and the other one with the Server. When it sends a message, first it encrypts the message with the key shared with the back-end and afterwards it encrypts the message again with the key shared with the gateway. When the gateway receives the packet, it decrypts it and transfers this packet to the Server. The gateway cannot access the confidential data, but it also can not translate the protocols between HTTP and WAP.

There exist two problems with ITLS. The first one is that the battery driven mobile phone needs to perform encrypting and decrypting of each packet twice: first with the key shared with the content provider and second with the key it shares with the WAP gateway. This implies more computational power is needed. The second problem occurs because ITLS must be deployed in the mobile phone as well as in the server. The server then needs additional mechanisms to distinguish whether the client uses TLS or ITLS.

IV. Security Considerations, Attacks and Threats

Based on the architectures as described in Section II, we identify two major security goals:

- Ensure E2E security between the end-hosts.
- Protect all the hosts in the LLN during an E2E security connection. This includes the protection of the LLN host and the LLN itself from (flooding, replay, amplification,

etc) attacks, the protection of the host in the Internet, as well as the protection of the 6LBR.

The LLN and the devices in the LLN are prone to flooding and resource exhaustion attacks because it consists of devices that are resource-constrained. We consider the attacker to be an Internet host which possesses the addresses of the target devices, and attempts to start multiple E2E handshakes with the LLN devices in order to disrupt the operation of the LLN. The DTLS handshake as it is, would allow anyone to initiate a handshake thus causing the victim device to repeatedly create a new DTLS context when a handshake request is received. Consequently, the attacker can exhaust the resources of battery-powered devices in the LLN. For that reason DTLS introduced a stateless cookie mechanism in order to prevent this attack. However, it is recognized that whenever the LLN device has to send a message, energy is consumed. Therefore, any undesirable and malicious network traffic could flood the network, utilizing not only the resources of the destination node, but also routing nodes and other nodes in the vicinity that receive the message on the physical layer.

CoAP messages usually contain commands to the nodes to do a specific task, e.g., 'Switch on the lights' or 'Send update messages'. Replaying these messages can disrupt the behaviour of the nodes, the 6LoWPAN and might lead to a complete system failure. Without filtering on the 6LBR, the replayed packet can greatly affect the performance of the network similar to the consequences of flooding attacks. DTLS has already defined a mechanism to detect message replay. However, before the node detects that a message is replayed, it has to receive, process and possibly forward the message between nodes. All these operations are energy consuming, and it has the same effect caused by a flooding attack. Given that the bandwidth in the LLN is limited, the fewer (unnecessary) packets are sent the better it is. Lastly, a mechanism to identify amplification attacks is needed. Amplified packets are quite huge, resulting in fragmentation on IP layer in the 6LBR, consuming energy in the receiving nodes on the path and the node itself by de-fragmenting the packet.

Based on these attacks and the architecture described in Section II, the challenge is on one hand to ensure E2E security between the end-devices and on the other hand to protect the LLN from attacks from the Internet (see [10], [11]). In both cases, the 6LBR plays an important role. As discussed in Section III, it is not desirable for the 6LBR to access the application data that is protected with a key shared between the back-end and the node. Consequently, the 6LBR is considered as an untrustworthy device at the application layer, which means that it does not have the key shared between back-end and the node. However, as the 6LBR is mediating the communication between the LLN and the Internet, and it is reasonable to assume some level of *trust at the network layer* on the 6LBR to forward the packets to the correct destination. In order to build such a *trust at the network layer*, mutual authentication with 6LBR and back-end is deemed necessary. With that, the 6LBR would only allow authenticated hosts in the Internet to communicate with the LLN, hence protecting LLN from attacks launched by attackers located in the Internet. Note that, in this case the 6LBR still has no access to the application data in the packets.

V. Approaches to E2E security

In order to avoid additional computations in the node, new mechanisms to protect the LLN from attacks should be established in the back-end and in the 6LBR. This involves also new functionalities for mapping TLS and DTLS in order to provide end-to-end security. In several use cases, deploying new protocols such as CoAP on the back-end service is not that straight forward and requires significantly more effort. For example, corporate systems running important services using legacy protocols that must be available at all times so that the service provisioning is not interrupted. Since it is complicated to upgrade these systems, an alternative is to have additional features installed on the end nodes supporting CoAP such that they play a role in the translation of the protocols. The new functions therefore must be lightweight and should consume as few resources as possible. The following subsections address approaches for both scenarios, (a) when the back-end can be equipped with new protocol features and (b) when the back-end cannot support CoAP.

A. Solutions for DTLS/DTLS and TLS/DTLS Scenarios with a back-end supporting CoAP

One approach to protect the LLN from attacks, launched from the Internet are tunnels (e.g., Virtual Private Networks (VPNs) [12]). The general idea behind an VPN is to interconnect two independent networks through a secure tunnel at network layer by hiding the actual receiver and the messages. The tunnel ends at the gateway (e.g., the 6LBR) in the other network. Thus, E2E connectivity needs additional mechanisms. Another issue is that this approach needs network support from the operating system, which is not always the case. Another method is to use secure tunnelling on transport layer [13]. A DTLS-DTLS tunnel can be established by first creating a DTLS channel from the back-end to the 6LBR. When successful, a second DTLS channel from the back-end to the node through the first channel is created (cf. [11]). For using the DTLS-DTLS tunnel approach, support from the operating system is not necessary but it requires changes in the network stack of the back-end system, as shown in Figure 2). This is also the case for a TLS-DTLS tunnel, created in the same way. Therefore, it is not suitable for back-end systems that do not support new protocols and changes in the network stack yet.

B. Solution for TLS/DTLS Scenarios with a back-end not supporting CoAP

One possibility to avoid changes in the back-end when it does not support CoAP, is to provide a direct mapping between TLS and DTLS in the 6LBR and the node. Table I summarizes the differences between the two protocols, TLS and DTLS when using PSKs. With this approach, the DTLS

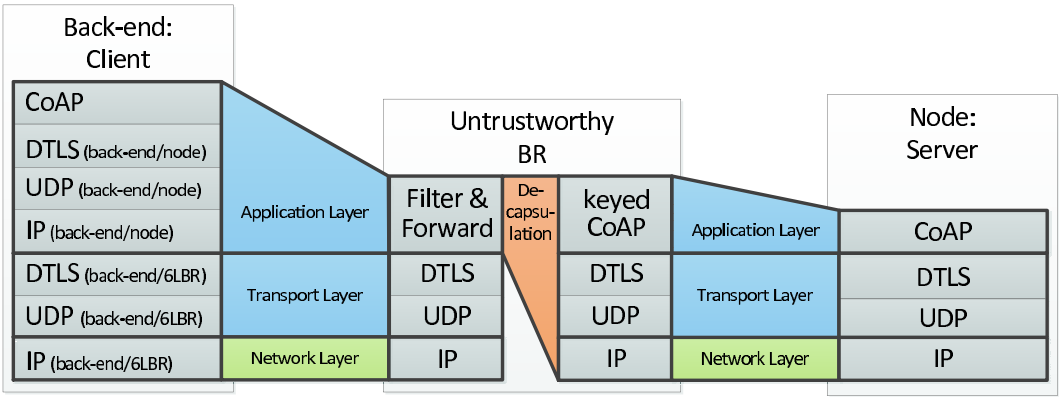

Figure 2. Message encapsulation and decapsulation in the DTLS-DTLS tunnel

cookie mechanism used for verifying the existence of inquiring client cannot be used. This is also not necessary because the TLS uses the TCP three-way handshake between back-end and 6LBR. The mapping consists of simply translating the fields of TLS to DTLS one by one. DTLS fields that do not exist in TLS are dropped from the message, however they are kept and tracked by the 6LBR and the node. When starting the communication, the node does not know whether the key establishment request originated either from a HTTP or a CoAP device. The 6LBR performs the translation of TLS to DTLS and vice versa in the first place. The verification of the FINISHED message (generated with the hash of the previously exchanged messages) will fail due to the changes that have been made. Figure 3 shows how the node can still

Record Header	TLS	DTLS
Content Type	(23) Application Data	
Version	TLSv1.2 {3, 3}	DTLSv1.2 {254, 253}
Epoch	not-existent	2 Byte
Sequence number	not-existent	6 Byte
Length during		
- Record process	Payload size + MAC size	
- Handshake	Message size + Handshake header size (5 Byte)	Message size + Handshake header size (13 Byte)
Handshake Header	**TLS**	**Comment**
Message Sequence	not-existent	Handled and monitored by the 6LBR
Fragment Offset		
Fragment Length		
CLIENTHELLO	**TLS**	**Comment**
Cookie	not-existent	Cookie mechanism cannot be used while mapping

Table I
MAPPING TLS AND DTLS

verify the correctness of the FINISHED message. This solutions requires an extension in the CoAP node. With this extension the verification of the FINISHED message is done twice in the back-end. The first FINISHED message is verified without any changes and will fail. The node can assume that the other party that it is communicating uses TLS and therefore translates all the previously exchanged messages to TLS and recalculates its value again. When it fails again, an attacker is recognized. Otherwise the node knows, that the back-end uses TLS for the communication. It creates its FINISHED message by translating the previously exchanged handshake messages to TLS so that the back-end can verify its correctness.

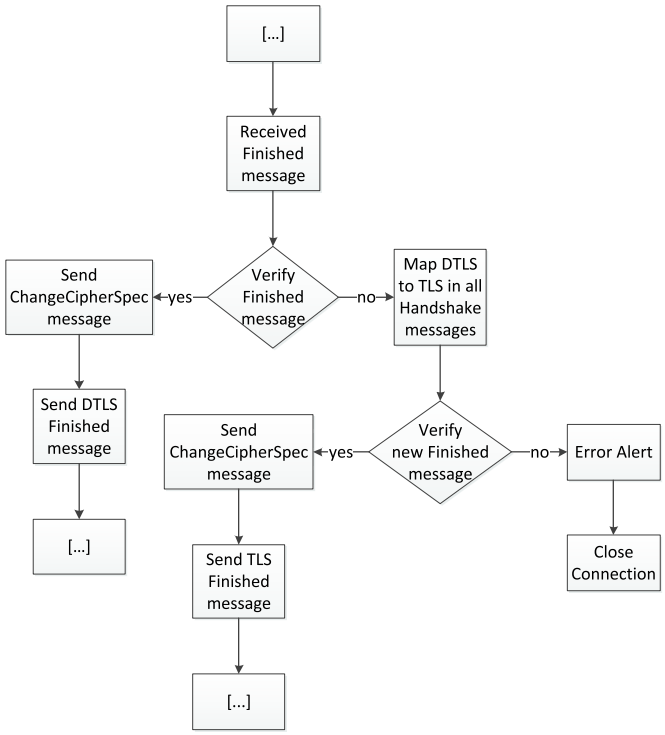

Figure 3. (D)TLS Finished message verification in CoAP node.

In the record layer of (D)TLS, a MAC (Message Authentication Code) is used for detecting replayed or tampered messages. It is created in the following way:

```
MAC(MAC_write_key, seq-nr,
    (D)TLS.type + (D)TLS.version
     + (D)TLS.length + (D)TLS.fragment)
```

The first value (`MAC_write_key`) is a key, shared between back-end and node after a successful handshake.

(D)TLS.type, (D)TLS.version and (D)TLS.length verify the correctness of the record header, whereas (D)TLS.fragment confirms the integrity of the actual message. The seq-nr is used in order to detect replay messages. As shown in Table I, the values for (D)TLS.version when using TLSv1.2 is {3, 3}, while when using DTLS in the same version {254, 253}. Also the sequence numbers in the TLS and DTLS MAC are different. The TLS sequence number starts from *zero* and with a size of 8 Byte, whereas the DTLS sequence number is the concatenation of the epoch (2 Byte) and the sequence number (6 Byte) from the DTLS record header. The epoch is increased by *one* after successful handshake. The computed MAC is added to the actual application data and the resulting message is encrypted with the session key, shared between the back-end and the node. When the back-end sends application data with DTLS, the 6LBR can only translate the unencrypted headers of the exchanged packets. The node verifies the MAC from a received message with TLS values, in case the handshake verification was successful with TLS content, otherwise the back-end uses DTLS and no translation is necessary.

VI. Analysis and Discussion

There still exist foreseeable concerns about the TLS-DTLS mapping approach. A big part of the mapping procedure is delegated to the node. This leads to extra source code and additional computational effort on the resource constrained device. But with this approach no extra protocols or cipher suite must be deployed in the node and no additional communication overhead is incurred in the constrained network. The mapping approach lacks of authentication and hence an attacker pretending to be the 6LBR is able to interfere with the communication. However, changes in the message mapping will still be recognized by node and back-end system but this leads to unnecessary battery consumption. Additionally, with this attack the aggregator is able to gather data from the headers for the purpose of cryptanalysis. In this setting, deploying a content filter firewall at the 6LBR could stop most DoS attacks involving TCP (e.g., TCP SYN flood attack, TCP reset attack, TCP sequence prediction attack). Thus, without authentication the 6LBR could still be useful for protecting the LLN to some extent, but this does not apply to all security threats.

When both the back-end and the node has successfully performed the handshake, the end-hosts share a secret session key. Message verification and replay detection cannot be done in the 6LBR due to the fact that it does not have access to the secret key. Therefore, this must be devolved to the CoAP device. For the same reason, it is apparent in the HTTP/CoAP use case that the HTTP-CoAP mapping cannot be done by the 6LBR, and a dedicated HTTP-CoAP proxy in the LLN needs to be deployed.

VII. Conclusion and Outlook

Applying existing Internet standards to smart devices simplifies the integration of the envisioned scenarios in the IoT. But as we have shown, security protocols such as TLS or DTLS that have been proven to be successful in the Internet does not necessarily mean that the same security levels can be achieved when interconnecting the Internet with the LLN. In this setting, new security issues get raised, and they need to be addressed.

Through our observation and analysis, providing E2E security in IoT is not so trivial, mainly due to many possible usage scenarios, i.e, CoAP/CoAP, DTLS/DTLS and HTTP/CoAP, TLS/DTLS mediated by a 6LBR, that have different constraints and requirements. Our analysis also reveals that having a secure E2E connection between two end hosts only provides a secure communication channel; the LLN can still be vulnerable to resource exhaustion, flooding, replay and amplification attacks, since the 6LBR typically does not perform any authentication.

We have described three security attacks that lead to resource exhaustion in the LLN and we have shown two approaches to mitigate such attacks. First, the mapping of TLS to DTLS protocol to ensure end-to-end security at the application layer, disallowing the 6LBR to gain access to the data in transit. Our goal is to optimize this approach in the future in order to mitigate the current drawbacks. As for the second, use of DTLS-DTLS tunnel to protect the LLN needs further investigation because having two independent tunnels would incur a lot of communication overhead.

References

[1] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained Application Protocol (CoAP)," IETF, Internet Draft, draft-ietf-core-coap-09 (work in progress), Mar. 2012.

[2] E. Rescorla and N. Modadugu, "Datagram Transport Layer Security version 1.2," IETF, RFC 6347, Jan. 2012.

[3] S. Kent and K. Seo, "Security Architecture for the Internet Protocol," IETF, http://tools.ietf.org/html/rfc4301, RFC 4301, Dec. 2005.

[4] N. Modadugu and E. Rescorla, "The Design and Implementation of Datagram TLS," in *In Proc. NDSS*, 2004.

[5] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet*. Wiley Publishing, 2010.

[6] WAP Forum, "WAP Architecture," online, http://www.mendeley.com/research/wap-architecture/, 2001.

[7] Phone.com, "Phone.com Secure Enterprise Proxy," pp. 3–4, 2000.

[8] S. Cobb, "The flip side of the wireless explosion: Dealing with WAP-gap security risks," online, http://cobbsblog.com/help/art-wap_gap.htm, 2001.

[9] E. Kwon, Y. Cho, and K. Chae, "Integrated transport layer security: end-to-end security model between WTLS and TLS," in *Proceedings of the The 15th International Conference on Information Networking*, ser. ICOIN '01. IEEE Computer Society, 2001.

[10] O. Garcia-Morchon, S. L. Keoh, S. Kumar, R. Hummen, and R. Struike, "Security considerations in the ip-based internet of things," IETF, Internet-Draft, draft-garcia-core-security-04 (work in progress), Mar 2012.

[11] M. Brachmann, O. Garcia-Morchon, S. L. Keoh, and S. Kumar, "Security Considerations around End-to-End Security in the IP-based Internet of Things," in *Workshop on Smart Object Security*, H. Tschofenig, Ed., March 2012. [Online]. Available: http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/

[12] C. Scott, P. Wolfe, and M. Erwin, *Virtual Private Networks*, ser. Animal Series. O'Reilly, 1999.

[13] R. Seggelmann, M. Tüxen, and E. Rathgeb, "Strategies to Secure End-to-End Communication - And Their Application to SCTP-Based Communication," Dec. 2012, to appear.