ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

DTLS & COAP Based Security For Internet of Things Enabled Devices

D.UthayaSinthan[*1], M.S.Balamurugan[2]
[*1,2] Communication Systems, SSIET, Coimbatore, India
duthaya20@gmail.com

Abstract

The Internet of Things (IoT) enables a wide range of application scenarios with potentially critical actuating and sensing function, e.g., in the e-health domain. For communication at the application layer, resource-constrained devices are expected to employ the Constrained Application Protocol (CoAP) that is currently being standardized at the IETF. To protect the transmission of sensitive information, secure CoAPmandates the use of Datagram TLS (DTLS) as the underlying security protocol for authenticated and confidential communication. DTLS, however, was originally designed for comparably powerful devices that are interconnected via reliable, high bandwidth links. In this work, we present Lithe – an integration of DTLS and CoAP for the IoT. With Lithe, we additionally propose a novel DTLS header compression scheme that aims to significantly reduce the energy consumption by leveraging the 6LoWPAN standard. Most importantly, our proposed DTLS header compression scheme does not compromise the end-to-end security properties provided by DTLS. At the same time, it considerably reduces the number of transmitted bytes while maintaining DTLS standard compliance. We evaluate our approach based on a DTLS implementation for the Contiki operating system. Our evaluation results show significant gains in terms of packet size, energy consumption, processing time, and network-wide response times when compressed DTLS is enabled.

Keywords: CoAP, DTLS, CoAPs, 6LoWPAN, Security, IoT

Introduction

The Internet of Things (IoTs) can be described as connecting everyday objects like smart-phones, Internet TVs, sensors and actuators to the Internet where the devices are intelligently linked together enabling new forms of communication between things and people, and between things themselves. Building IoTs has advanced significantly in the last couple of years since it has added a new dimension to the world of information and communication technologies. With the advancements in Internet technologies and Wireless Sensor Networks (WSN), a new trend in the era of ubiquity is being realized.The 6LoWPAN concept originated from the idea that the Internet Protocol could and should be applied even to the smallest devices, and that low-power devices with limited processing capabilities should be able to participate in the Internet of Things. The 6LoWPAN group has defined encapsulation and header compression mechanisms that allow IPv6 packets to be sent to and received from over IEEE 802.15.4 based networks. IPv4 and IPv6 are the work horses for data delivery for local-area networks, metropolitan area networks, and wide-area networks such as the Internet. Likewise, IEEE 802.15.4 devices provide sensing communication-ability in the wireless domain. Such IP-connected smart devices (Things) are becoming part of the Internet hence forming the Internet of Things (IoT) or strictly speaking the IP-connected IoT. To cope with constrained resources and the size limitations of IEEE 802.15.4-based networks2, 6LoWPAN header compression mechanisms are defined. The 6LoWPAN standard already defines the header compression format for the IP header, IP extension headers, and the UDP header.

The target for IP networking for low-power radio communication are the applications that need wireless internet connectivity at lower data rates for devices with very limited form factor. Examples could include, but are not limited to: automation and entertainment applications in home, office and factory environments. The header compression mechanisms standardized in RFC6282 can be used to provide header compression of IPv6 packets over such networks.IPv6 is also in use on the smart grid enabling smart meters and other devices to build a micro mesh network before sending the data back to the billing system using the IPv6 backbone. Some of these networks run over IEEE 802.15.4 radios, and therefore use the header compression and fragmentation as specified by RFC6282.

ISSN: 2277-9655
Impact Factor: 1.852

Security Aspects

Providing E2E security is a widely explored area in conventional Internet communication. However, there has been comparatively less research conducted in E2E security considering 6LoWPANs. The resource constraints of the devices and the lossy nature of wireless links are among the major reasons that hinder applying general E2E security mechanisms to 6LoWPANs. Recently, the community has presented works on analysing security challenges in the IP-based IoT and solutions that improve or modify standard IP security protocols for the requirements of resource-constrained devices. In our discussion of related work, we focus on approaches that aim to enable E2E security solutions in the IoT. IPsec security services are shared among all applications running on a particular machine. Even though our 6LoWPAN compressed IPsec can be used to provide lightweight E2E security at the network layer, it is not primarily designed for web protocols such as HTTP or CoAP. For web protocols TLS or DTLS are common security solutions. TLS works over TCP, whereas in 6LoWPAN networks UDP is preferred.

Datagram Transport Layer Security (DTLS)

The DTLS protocol provides communications privacy for datagram protocols. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery. The DTLS protocol is based on the Transport Layer Security (TLS) protocol and provides equivalent security guarantees. Datagram semantics of the underlying transport are preserved by the DTLS protocol. DTLS is a derivation of SSL protocol. It provides the same security services (integrity, authentication and confidentiality) but under UDP protocol. DTLS is implemented by several projects including SSL and the OpenSSL project.UDP: Typically DTLS uses UDP as its transport protocol. There is no well-known UDP port for DTLS traffic.

Related Work

IPv6 over Low power Wireless Personal Area Network (6LoWPAN) enables the use of IP in low-power and lossy wireless networks such as Wireless Sensor Networks (WSNs). Such IP-connected smart devices (Things) are becoming part of the Internet hence forming the Internet of Things (IoT) or strictly speaking the IP-connected IoT. To cope with constrained resources and the size limitations of IEEE 802.15.4-based networks2, 6LoWPAN header compression mechanisms are defined. The 6LoWPAN standard already defines the header compression format for the IP header, IP extension headers, and the UDP header. To believe it is particularly beneficial to apply the 6LoWPAN header compression mechanism to compress other protocols having well-defined header fields, such as DTLS. This work provides a lightweight CoAPs by compressing the underneath DTLS protocol with 6LoWPAN header compression mechanisms. The purpose of DTLS header compression is twofold. First, achieving energy efficiency by reducing the message size, since communication requires more energy than computation.Second, avoiding 6LoWPAN fragmentation that is applied when the size of datagram is larger than the link layer MTU. Our compressed DTLS maintains true End-to-End (E2E) security between Lithe enabled hosts in 6LoWPAN networks and typical Internet hosts that use uncompressedCoAPs. In our previous work, they propose a header compression method to use IPsec to secure the communication between nodes in 6LoWPAN networks and hosts in the Internet. To define Next Header Compression (NHC) encodings to compress the Authentication Header (AH) and Encapsulating Security Payload (ESP) extension headers. Even though our 6LoWPAN compressed IPsec can be used to provide lightweight E2E security at the network layer, it is not primarily designed for web protocols such as HTTP or CoAP.

CoAP and DTLS

CoAP is a web protocol that runs over the unreliable UDP protocol and is designed primarily for the IoT. CoAP is a variant of the most used synchronous web protocol, HTTP, and is tailored for constrained devices and machine-to-machine communication. However, whileCoAP provides a REST interface similar to HTTP, it focuses on being more lightweight and cost-effective than its variant for today's Internet. To protect CoAP transmissions, Datagram TLS (DTLS) has been proposed as the primary security protocol [2]. Analogous to TLS-protected HTTP (HTTPs), the DTLS-secured CoAPprotocol is termed CoAPs. DTLS guarantees E2E security of different applications on a single machine by operating between the transport and application layers. DTLS consists of two layers: the lower layer contains the Record protocol and the upper layer contains either of the three protocols namely Handshake, Alert, and ChangeCipherSpec, or application data. The ChangeCipher-Spec is used during the handshake process to merely indicate that the Record protocol should protect the subsequent messages with the newly negotiated cipher suite and security keys. DTLS uses the Alert protocol to communicate theerror messages between the DTLS peers. Figure 2 shows the structure of a DTLS message in an IP/UDP datagram. The Record protocol is a carrier for the upper layer protocols. The Record header contains among others contenttype and fragment fields. Based on the value in the content type, the fragment field contains either theHandshake protocol,

Alert protocol, ChangeCipherSpec protocol, or application data.The Record header is primarily responsible to cryptographically protect the upper layer protocols or application data once the handshake process is completed. The Record protocol's protection includes confidentiality,integrity protection and authenticity. The DTLS Record is a rather simple protocol whereas the Handshake protocol is a complex chatty process and contains numerous message exchanges in an asynchronous fashion.Figure 3 shows a full handshake process. The handshake messages, usually organized in flights, are used to negotiate security keys, cipher suites and compression methods. Thescope of this paper is limited to the header compression only and not the cryptographic processing of Record and Handshake protocols.

Proposed System

Implementation

Our proposed header compression mechanisms in Lithe can be implemented in any OS that supports 6LoWPAN.The Lithe implementation consists of four main components: (i) DTLS, (ii) CoAP, (iii) CoAPDTLS integration module, (iv) DTLS header compression. For DTLS we use the open source tinyDTLS implementation which supports the basic cipher suite based on pre-shared key TLS_PSK_WITH_AES_128_CCM_8. We adapt tinyDTLS for the WiSMote platform and for the 20-bit address support of msp430-gcc(version of 4.7.0). For CoAP, we use the default CoAP implementation in the Contiki OS. We develop the integration module that connects the CoAP and DTLS implementations and enables the CoAPs protocol. This integration allows the application independent access to CoAPs where outgoing CoAP messages are transparently handed to DTLS that transmits the protected messages to the destination. All incoming CoAP messages are protected through DTLS and therefore are processed first at the DTLS layer and handed transparently to CoAP, which resides in the application layer. We implement our proposed header compression as an extension to the 6LoWPAN implementation in the Contiki OS. The 6LoWPAN layer resides between the IP and Medium Access Control (MAC) layers. The packets from the IP layer that are ready to be transmitted from the node are considered as output packets. The packets from the MAC layer that are received to the node are considered as input packets. The 6LoWPAN layer processes all UDP packets from both directions. Therefore, we use two ways to distinguish UDP packets that carry DTLS messages as payload from other UDP packets. In the case of input packets, the pre-configured default DTLS port is used to identify CoAPs messages. In the second case when the packet is received from the MAC layer, the DTLS port and the ID bits in the NHC-for-UDP and in the NHC for DTLS headers are used to distinguish the compressed headers from the uncompressed. Details are provided in Section IV. Furthermore, it is important to emphasize, that while applying header compression, the E2E security of DTLS is not compromised. This is due to the design of DTLS and our effort to remain standard-compliant. The header fields are, after final negotiation of the cipher suite, integrity protected within the Record layer. During the compression/decompression process the original headers are not modified and the integrity protection is maintained. After decompression in the 6LoWPAN layer, the integrity of the packet is checked in the DTLS layer. The correctness of integrity protection serves as well as a proof of correct decompression.

Evaluation

On real sensor nodes running the ContikiOS. We use WiSMoteas our hardware platform. WiSMotes are equipped with *(i)* a 16 MHz, MSP430 5-Series, 16-bit RISC microcontroller, *(ii)* 128/16 kB of ROM/RAM, and *(iii)* an IEEE 802.15.4 (CC2520) transceiver. We select WiSMotes because of the RAM and ROM requirements of the DTLS implementation, which is discussed in more detail in Section VI-B. The network setup consists of two WiSMotes which communicate directly through the radio. The CC2520 transceiver provides an AES-128 security module. However, for our evaluation we do not use the AES hardware support and rely on software AES. Leveraging the AES hardware support for the cryptographic computations involved in DTLS would lead to higher performance. The focus of our evaluation is on the impact of DTLS header compression on response time and energy consumption of nodes. Therefore, the performance loss due to software AES is not affecting our evaluation. Furthermore, we do not enable link layer security support, in order to be able to analyze the processing overhead of compression separately. In our previous work, we have evaluated the performance gains when using the AES support in hardware. There, we implement and evaluate the IEEE 802.15.4 link layer security.

Experimental Results

1) DTLS Compression Overhead: The overhead caused through in-node computation for compression and decompression of DTLS headers is almost negligible. However, we measure and show it for the sake of completeness. Figure 8 shows the additional energy consumed for compression(compressing/decompressing) of the handshake messages. Each handshake message consists of the both Record and Handshake headers. For a DTLS handshake based on preshared keys, on average, 4.2 uJ of energy is consumed for compression.

ISSN: 2277-9655
Impact Factor: 1.852

2) CoAPs Initialization: During the CoAPs initialization phase a secure session is established between the two communicating end-points using the DTLS handshake protocol. The handshake process uses both the Record and Handshake headers, which means that both of these headers can be compressed. The tradeoff between additional in-node computations vs. reduced packet sizes shows itself in the energy consumption for packet transmission in a DTLS handshake.Table III compares the energy consumption required for transmission for the case compression is applied and respectively for the case, where compression is not applied. On average 15% less energy is used to transmit (and receive) compressed packets. This is due to smaller packet sizes achieved through compression.

3) CoAPs Request-Response: Once the CoAPs initialization phase is completed, i.e., the handshake has been performed, a sensor node can send/receive secure CoAP messages using the DTLS Record protocol. Although the Handshake protocol is, compared to the Record protocol, a more resource hungry protocol, it is performed only once during the initialization phase and/or later (rarely) for re-handshake.

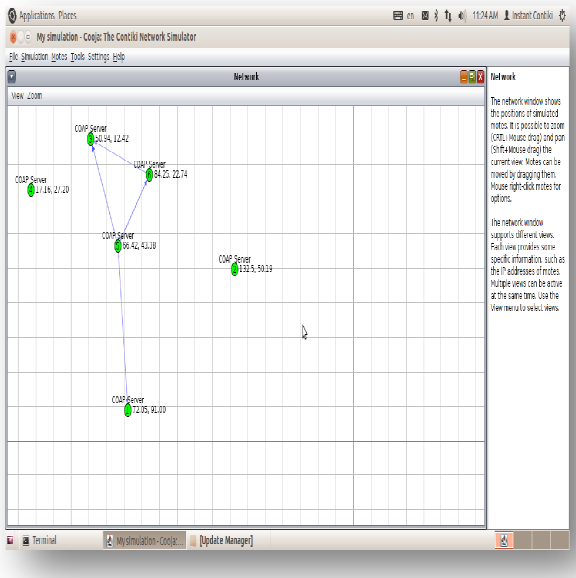

Fig 1CoAP output in Network Simulator

In order to measure the performance of compression of the Record Header, we measure the energy consumption and the round trip time (RTT) for the processing of CoAP requestresponse messages. We start our measurements when the client prepares the CoAP request, and stop after the server's response is received and processed. The corresponding CoAP response contains varying payload lengths. To be more precise, eight different payload sizes in the range of 0 to 48 bytes are used. We select 48 bytes, because with 48 byte CoAP payload 6LoWPAN fragmentation is performed in case of plain CoAPs. The transmission of CoAPGET requests has the same amount of energy consumption since the size of request messages are always constant. Hence, energy consumption for CoAPs requests is always reduced by 10% using compression. The energy savings for the CoAPs response messages depend on the payload length and whether compression can prevent fragmentation. The latter is the case for a payload length of 48 byte. Hence, the energy saving is in the range of 4-26%, where the highest energy saving is for 48 byte. For analyzing the overall energy consumption savings for CoAPs request-responses, we sum up energy consumption for packet transmission on the server and client. We observe that in average energy savings of about 7% are achieved. However, in the case where fragmentation is avoided through compression, the savings increase to 20.6%.

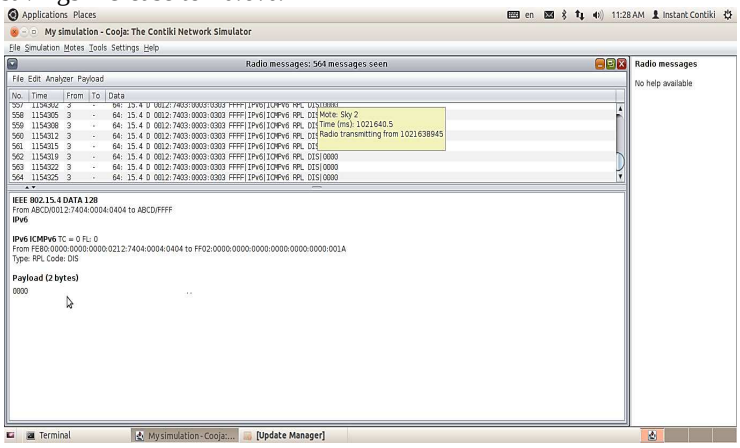

Fig 2Compressed CoAP function

This is due to the fact, that with 48 byte payload, 6LoWPAN transmits the packet within two fragments, whereas with compression the packet is transmitted without fragmentation. The reduced transmission time affects as well the RTT for a CoAPs request-response message. In the case of no RDC, as shown in Figure 10b, the RTT is in average 1.5% smaller, except for 48 byte payload. There, the RTT with compression is even 77% smaller, since fragmentation is avoided. In order to assess the overall overhead caused through security, we have as well added values forCoAP without security. The RTT in CoAP without security is in average 1/3 of the CoAPs, as long as no fragmentation is needed. Looking at the RTT with RDC, as shown in Figure 10b we see that for all three cases of: (i) CoAP without any security, (ii) plain CoAPs, and (iii) CoAPs with DTLS compression (Lithe), RTT values are in the same range, expect for CoAP response messages with 48 Byte payload. This is a side-effect of RDC. RDC saves

energy by putting the radio into sleep for the most of the time. However, this happens at the cost of higher latency. Packets in RDC networks are not transmitted directly. For example, in Figure 10b for 48 byte payload, compression leads to 50% shorter RTT.

Discussion and Future Work

CoAP enabled hosts will be an integral part of the Internet of Things (IoT). Furthermore, real world deployments of CoAP enabled devices require security solutions. To this end, DTLS is the standard protocol to enable secure CoAP (CoAPs). In this paper, we investigate the possibility of reducing the overhead of DTLS by means of 6LoWPAN header compression, and present the first DTLS header compression specification for 6LoWPAN. We quantitatively show that DTLS can be compressed and its overhead is significantly reduced using 6LoWPAN standardized mechanisms. Our implementation and evaluation of compressed DTLS demonstrate that it is possible to reduce the CoAPs overhead as the DTLS compressions efficient in terms of energy consumption and network wide response time, when compared with plain CoAPs. The difference between compressed DTLS and uncompressed DTLS is very significant, if the use of uncompressed DTLS results in 6LoWPAN fragmentation. As future work we plan to deploy Lithe in a real world IoT system with a real application scenario. Such an IoTsetup consists of constrained devices, standard computers, and smartphones. A real world deployment helps us to thoroughly evaluate Lithe in an heterogeneous IoT, and ultimately demonstrate the use of Lithe in security sensitive applications.

References

[1] J. Hui and P. Thubert.Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. RFC 6282, RFC Editor, September 2011.

[2] Z. Shelby, K. Hartke, C. Bormann, and B. Frank. Constrained Application Protocol (CoAP). Internet-Draft draft-ietf-core-coap-16.txt, IETF Secretariat, May 2013.

[3] E. Rescorla and N. [Modadugu. Datagram Transport Layer Security Version 1.2. RFC 6347, RFC Editor, January 2012.

[4] R. Hummen, J. Hiller, H. Wirtz, M. Henze, H. Shafagh, and K. Wehrle.6LoWPAN Fragmentation Attacks and Mitigation Mechanisms. In Proceedings of the 6th ACM Conference on Security and Privacy inWireless and Mobile Networks (WiSec), April 2013.

[5] A. Dunkels, B. Gr ¨onvall, and T. Voigt.Contiki - a lightweight and flexible operating system for tiny networked sensors. In EMNets'04, Tampa, USA, November 2004.

[6] T. Heer, O. Garcia-Morchon, R. Hummen, S. Keoh, S. S. Kumar, and K. Wehrle. Security Challenges in the IP-based Internet of Things. Springer Wireless Personal Communications Journal, 2011.

[7] S. Raza, S. Duquennoy, A. Chung, D. Yazar, T. Voigt, and U. Roedig.Securing communication in Systems(DCOSS'11), Barcelona, Spain, 2011.

[8] J. Granjal, E. Monteiro, and J. S. Silva.Network-layer security for the Internet of Things using TinyOS and BLIP.International Journal ofCommunication Systems, 2012.

[9] M. Brachmann, S. L. Keoh, O. G. Morchon, and S. S. Kumar. End-toend transport security in the IP-Based Internet of Things. In ComputerCommunications and Networks (ICCCN), 2012 21st International Conference on, pages 1 –5, August 2012.

[10] T. Kothmayr, C. Schmitt, W. Hu, M. Brunig, and G. Carle. A DTLS based end-to-end security architecture for the Internet of Things with two-way authentication. In Local Computer Networks Workshops, 2012IEEE 37th Conference on, pages 956–963. IEEE, 2012.

[11] J. Granjal, E. Monteiro, and J. S. Silva.On the feasibility of secure application-layer communications on the Web of Things.In LocalComputer Networks (LCN), 2012 IEEE 37th Conference on, pages 228 –231, October 2012.

[12] S. Keoh, S. Kumar, and O. Garcia-Morchon.Securing the IP-based Internet of Things with DTLS.Working Draft, February 2013.

[13] R. Hummen, J. H. Ziegeldorf, H. Shafagh, S. Raza, and K. Wehrle.Making Certificate-based Authentication Viable for the Web of Things.In Proceedings of the 2nd ACM Workshop on Hot Topics on WirelessNetwork Security and Privacy (HotWiSec), April 2013.

[14] C. Bormann. 6LoWPAN Generic Compression of Headers and Headerlike Payloads.Internet-Draft draft-bormann-6lowpan-ghc-04.txt, IETF Secretariat, March 2012.

[15] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs):

ISSN: 2277-9655
Impact Factor: 1.852

Overview, Assumptions, Problem Statement, and Goals. RFC 4919, RFC Editor,August 2007.
[16] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550, RFC Editor, March2012.