LWIG Working Group
Internet-Draft
Intended Status: Informational
Expires: February 27, 2014

S. Keoh
University of Glasgow
S. Kumar
O. Garcia-Morchon
Philips Research

August 27, 2013

Securing the IP-based Internet of Things with DTLS
draft-keoh-lwig-dtls-iot-02

Abstract

The IP-based Internet of Things (IoT) refers to the pervasive interaction of smart devices and people enabling new applications by means of IP protocols. Traditional IP protocols will be further complemented by 6LoWPAN and CoAP to make the IoT feasible on small devices. Security and privacy are a must for such an environment. Due to mobility, limited bandwidth, resource constraints, and new communication topologies, existing security solutions need to be adapted. We propose a security architecture for the IoT in order to provide network access control to smart devices, the management of keys and securing unicast/multicast communication. Devices are authenticated and granted network access by means of a pre-shared key (PSK) based security handshake protocol. The solution is based on Datagram Transport Layer Security (DTLS). Through the established secure channels, keying materials, operational and security parameters are distributed, enabling devices to derive session keys and group keys. The solution relies on the DTLS Record Layer for the protection of unicast and multicast data flows. We have prototyped and evaluated the security architecture. The DTLS architecture allows for easier interaction and interoperability with the Internet due to the extensive use of TLS. However, it exhibits performance issues constraining its deployment in some network topologies and hence would require further optimizations.

Status of this Memo

Copyright and License Notice

Table of Contents

1 Introduction

The IP-based Internet of Things (IoT) will enable smart and mobile devices equipped with sensing, acting, and wireless communication capabilities to interact and cooperate with each other in a pervasive way by means of IP connectivity. IP protocols play a key role in this vision since they allow for end-to-end connectivity using standard protocols ensuring that different smart devices can easily communicate with each other in an inexpensive way. Protocols such as IPv6, TCP and HTTP that are commonly used in traditional networks will be complemented by IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) and Constrained Application Protocol (CoAP) currently in development in IETF.

This allows smart and mobile devices used for various applications like healthcare monitoring, industrial automation and smart cities to be seamlessly connected to the Internet, thus creating a plethora of IoT applications. An example application is smart-metering, in which a smart-meter can communicate with consumer electronics and other devices in a building/household to retrieve and manage energy consumption. Additionally, a set of lighting devices could be controlled and managed efficiently by the smart-meter, e.g., dimming them down during peak energy periods by means of a multicast message.

Security and privacy are mandatory requirements for the IP-based IoT in order to ensure its acceptance. The interaction between devices must be regulated in the sense that authorized devices joining a specific IoT network in a given location will be granted access to only certain resources provided by the IoT.

To enable this, the IoT network has to:

1. Authorize the joining of the smart device, such that it is provisioned and configured with the corresponding operational parameters, thus providing "network access".
2. Establish and derive pairwise keys, application session keys and multicast keys to enable devices to secure its communication links with each other, and for that, "key management" is needed.
3. Devices should be able to communicate within the network, either securely pairwise or in a "secure multicast" group.

In order to achieve these three security functionalities, there are several challenges: (i) no standard solution exists yet; (ii) mobility of smart devices should be accounted for; (iii) the solution needs to be applicable to large scale deployments; (iv) new communication patterns introduced in IoT such as multicast (beyond just end-to-end communication links), and thus, IP security protocols need to be adapted; and (v) the available resources (bandwidth,

memory, and CPU) are tightly constrained.

Of course, the three research topics are not new, and indeed, some of them (e.g., key management or secure broadcast) have been extensively analyzed in the wireless sensor network literature during the last decade. However, the last step of applying those results to actual standards to get a working solution is still a missing and crucial step towards the success of the IP-based IoT. This work analyzes how this can be achieved.

We present a security architecture for the IP-based IoT in order to explore how these functionalities can be achieved by adapting and extending IP security protocols. With this, our goal is to analyze the trade-offs regarding performance, security, and interoperability so that we obtain a solution that performs reliably, offers high security, and is as interoperable as possible with the standard Internet.

The solution is based on Datagram Transport Layer Security (DTLS). We use the DTLS handshake for network access. For key management, we integrate with the Adapted Multimedia Internet KEYing (AMIKEY) protocol for efficient key management and generation of pairwise keys within an IoT network. Secure multicast operation is enabled through the direct use of DTLS record layer with the multicast keys to protect CoAP messages on top of IP multicast.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This Internet Draft defines the following terminology:

Internet of Things (IoT): A paradigm in which a diverse set of devices with different resources and capabilities (including sensors, actuators, smart phones, etc) are connected to the Internet, each is equipped with a uniquely identifiable IP address that can be contacted from anywhere and at anytime.

IoT domain: An IoT network that is connected to the public Internet through a number of 6LoWPAN border routers where the devices and services in the network are managed by a domain manager that could be located within the IoT network itself or in the public Internet.

Network access: A joining device is authenticated and then checked whether it is authorized to join a network. An IP address and a link-layer (L2) key are allocated to the joining device upon successful

authentication and authorization of the joining device, hence enabling the device to communicate in the secure network. This process is called network access.

Key management: A process of distributing, updating and renewing cryptographic materials including keying materials for deriving unicast and multicast keys, random numbers, session keys for unicast communication, and multicast group keys.

Security handshake protocol: A security protocol to authenticate two communicating devices and subsequently establishes a shared secret-key between them to secure their communication. The Datagram Transport Layer Security (DTLS) is referred as the security handshake protocol in this specification.

Link local address: A stateless IPv6 address that is intended for a point-to-point communication between two devices that are within the communication of each other. The packets with a link local address will not be routed or forwarded further by routers.

Pairwise key: A secret symmetric key that is shared between two communicating devices in the network, enabling them to encrypt and authenticate data packets exchanged between them.

Multicast key: A secret symmetric key that is shared by a group of devices in the network. It is used to protect the multicast group communication.

2. Related Work and Background

The "Datagram Transport Layer Security (DTLS)" protocol [RFC4347] is a datagram-compatible adaptation of TLS that runs on top of UDP. DTLS uses similar messages as defined in TLS including the DTLS handshake to establish a secure unicast link and the DTLS record layer to protect this link. The "DTLS handshake" supports different types of authentication mechanisms, e.g., using a pre-shared key, public-key certificates, and raw public-keys. DTLS is the mandatory standard for protection of CoAP [I-D.ietf-core-coap] messages. DTLS differs from TLS mainly in three aspects:

(1) DTLS provides DoS protection through a stateless cookie exchange;
(2) DTLS adds functionality to ensure a reliable link during its handshake to solve UDP's inherent packet losses and reordering;
(3) The record layer includes an explicit sequence number (again, due to the reordering issues in UDP) so that payload integrity and reply protection can be ensured.

The Adapted Multimedia KEYing (AMIKEY) [I-D.alexander-roll-mikey] is

used to provide keying material for securing uni- and multicast communications within constrained networks and devices. It is based on MIKEY, a Key Management Protocol intended for use in real-time applications [RFC3830]. For this purpose, AMIKEY provides different message exchanges that may be transported directly over UDP and TCP. Essentially, they can be integrated within other protocol like DTLS. Our solutions make use of AMIKEY's key derivation mechanism as we consider it to be efficient for constrained networks.

3 Use Cases & Problem Statement

```
                                       +------------------------------+
                                       |                +======+      |
                                     +-|-->6LBR         |Light1|      |
                                     | |                +======+      |
                                     | |           +======+           |
                                     | |           |Light2|  +======+ |
                                     | |           +======+  | Win2 | |
                                     | | Floor 2             +======+ |
    +---------+   Internet   +-----+ | +------------------------------+
    | Backend | <----------> | BMS |-+
    +---------+              +-----+ | +---------------------------+
                                     | |                 +======+  |
                                     +-|-->6LBR          |Light4|  |
                                       |      +======+   +======+  |
                                       |      |Light3|             |
                                       |      +======+   +======+  |
                                       |                 | Win1 |  |
                                       | Floor 1         +======+  |
                                       +---------------------------+
```

Figure 1: Building Management Systems (BMS) Scenario

Our work targets an "IoT network" running 6LoWPAN/CoAP over multiple hops with both uni- and multicast links, i.e., typical edge networks in the IoT. Devices in the "IoT network" are mobile or stationary and exhibit tight processing, memory and bandwidth constraints. The "IoT network" is connected to the public Internet through a number of 6LoWPAN border routers (6LBR). Further, we consider a centrally managed scenario in which the devices and services in each "IoT network" are managed by a "domain manager". The "domain manager" could be located within the "IoT network" or in the public Internet. The "domain manager" along with the "IoT networks" it manages is denoted as the "IoT domain". Figure 1 illustrates an example of Building Management Systems (BMS) scenario where smart devices within the building (e.g., lighting devices, window blinds) form several multi-hop IoT networks connected to a remote building management

system via some border routers.

3.1 Problem Statement and Requirements

We consider an IoT domain with many devices that dynamically join the network, then provide or request a certain service and finally leave the network. These services are provided or requested using either unicast (e.g., switching on the heater) or multicast group communication (e.g., switching on all lights in a room). We identify three main problems that currently lack a standardized solution for IoT networks:

- Network access -- A new joining device must only be able to communicate in a secure IoT network after securely joining the IoT network and receiving all necessary access parameters, e.g., commissioning a new lighting bulb into the building network.

  The multi-hop nature of IoT networks leads to a key challenge here since a joining device and the domain manager cannot reach each other by means of regular IP routing so that specific approaches are needed.

  Similarly, devices that leave the IoT network should not be able to access the network with previous access parameters.

- Key management -- A lightweight mechanism to derive and manage different keys to secure interactions in the IoT domain is required, e.g., different pairwise, group, and network keys.

- Secure uni- and multicast communication -- A secure transport protocol is needed to protect the communication in the IoT domain.

  This includes both unicast and multicast links, protected using the derived keys, e.g., preserving the integrity and confidentiality of the exchanged commands.

Additional requirements are that the solutions need to be scalable for an IoT domain with several hundreds or thousands of resource constrained devices and be based on standard IP protocols for easier interaction and interoperability with the Internet. In this work, we provide a solution to these three problems based on a smart combination of DTLS and AMIKEY with only minimal modifications.

3.2 Threat model, Security Goals & Assumptions

We assume the Internet Threat Model [RFC3552] in which a malicious

adversary can read and forge network traffic between devices at any point during transmission, but assume that devices themselves are secure. In many IoT application areas the network is indeed untrusted (e.g., wireless communications in public places, large factories, office buildings). Security of the end devices is important to create a secure IoT scenario, however device security is not within the scope of this draft. The Internet Threat Model is thus a reasonable choice in our context.

We further identify the following threats in an IoT domain and the corresponding security goals:

- Secure Network Access -- Attackers can perform network attacks, e.g., flooding the network and using the network for other communication purposes. To this end, only devices that have been authenticated and authorized through a secure network access process should be allowed to communicate within the network.

- Key Management -- Attackers can attempt to compromise the keying materials, pairwise keys, or multicast group keys by exploiting the vulnerability on the devices. Therefore, secure key derivation and key update mechanisms are required to manage all cryptographic keys. Similarly, compromising the derived keys does not enable the attackers to obtain information about the keying materials.

- Secure Uni- and Multicast -- The adversary can maliciously modify either unicast or multicast traffic in the IoT network. Additionally, the adversary can eavesdrop on the data exchanged within an IoT domain. For unicast, two communicating parties must establish a pairwise key to secure the confidentiality, integrity and authenticity of the information exchanged. Multicast communication is protected using a group key, thus only allowing authenticated and authorized group members to send messages to a multicast group. We assume that the multicast group members can trust each other since they are authenticated and authorized by the domain manager. Therefore, we do not additionally require source authentication in the messages as will be detailed further later on. If a device leaves a multicast group, it must not be able to rejoin and send messages to the group later on.

Due to the resource constrained devices in the IoT network, our proposed security architecture is based on the assumption that a device has been configured with a PSK that is known "a priori" to the domain manager of the IoT domain it wants to join. This assumption is reasonable since a PSK could be embedded and registered during the manufacturing process of a device and the domain manager can retrieve

it from a central server. If more powerful devices are available, our secure network access can be easily updated to work with public-key cryptography.

4 Design

In this section, we detail the design of our solution to the three problems (i) secure network access, (ii) key management, and (iii) uni- and multicast communication as identified in Section 3.1. The solution is mainly built on DTLS and AMIKEY.

4.1 Overview

The first phase accounts for "secure network access". In our architecture, the network is protected at link layer by means of a symmetric-key (L2 key), which is unknown to the joining device "a priori". Using its link local address, the joining device authenticates itself to the domain manager of the IoT domain by means of an initial handshake (DTLS) that is based on a PSK. The PSK is assumed to have been pre-configured in the device (cf. Section 3.2). On success, the domain manager issues access parameters (L2 key) that would allow the joining device to access the secured IoT network and to receive a routable IPv6 address. As the link local address only enables one hop communication, this poses a key challenge in multi-hop networks in that the domain manager cannot be reached directly. In the proposed solution, this issue is dealt with by means of a relay device, e.g., as was done in the PANA protocol by means of the PANA relay element [RFC6345]. Figure 2 shows the generic logic of the relay element.

```
          +--------------+ N  +--------------------------+ Y  +-------+
    (S)-->|L2 secure msg?|--->| Network Access Request   |--->|Process|
          +--------------+    | msg & link-local Address?|    |& Relay|
                 |            +--------------------------+    +-------+
                 |Y                         |
                 |                          |N
          +-----------------+               |
          |Normal processing|            +------+
          +-----------------+            | Drop |
                                         +------+
```

Figure 2: Relay logic for secure network access

The second phase deals with "key management". The joining device is provided with keying material to interact with other devices either in pairs or groups. For pairwise key generation, two communicating devices that wish to interact with each other can derive a pairwise

key based on their identities, e.g., using a polynomial scheme [Blundo-polynomial]. For group key generation, we assume that the domain manager controls the multicast groups. A joining device that wishes to participate in a multicast group indicates this to the domain manager during the initial handshake, and if authorized, receives the required multicast group keys from the domain manager.

Neither pairwise nor group keys are used directly, instead they serve as root keying material in the MIKEY key derivation mechanism in order to derive fresh purpose-specific session keys for any pair or group of devices in the IoT network. The protocol framework for requesting and managing these purpose-specific keys is based on the lightweight MIKEY-extension called AMIKEY [I-D.alexander-roll-mikey].

The final phase, "secure uni- and multicast", is achieved by using CoAP carried over the DTLS record layer. The pairwise or group keys derived in the key management phase are used to protect the communication links.

4.2 Secure Network Access

We describe the details of the initial DTLS based handshakes as well as how multi-hop environments are handled.

IETF CoRE working group defines DTLS for securing the transport of messages in an IoT domain. In this approach, the DTLS handshake protocol is used during the secure network access phase. The DTLS might be based on public-key certificates or raw public-keys as specified in CoAP. Our design uses DTLS-PSK [RFC4279], because it incurs less overhead and reduces the number of exchanged messages. We now describe how DTLS-PSK is used in a single- and multi-hop scenario.

> Single-hop: In this case, the joining device can be authenticated with the domain manager by performing the DTLS handshake based on the PSK and relying on the link-local address. Once the device has been authenticated and authorized for the network, the established DTLS secure channel between the domain manager and the joining device is used to issue the L2 key to the device. DTLS-PSK provides resiliency against Denial-of-Service attacks through a cookie mechanism.
>
> Multi-hop: DTLS runs on UDP but communication is limited to a single hop due to the link local address. To deal with this, a relay device is responsible for forwarding the messages by using a mapping between the link local address of the joining device and the relay's IP address. The relaying logic consists of changing the link local address of the joining device with

the relay device's IP address, in a similar way as done in PANA relay element [6345]. This indeed allows for the provisioning of L2 key to the joining device so that it can later receive its IP address through the neighbor discovery protocol [6775]. However, note that the DTLS channel established during the handshake is bound to the relay's IP address and not the new IP address of the joining device. Hence, if the device were to communicate with the domain manager again, it would have to redo the DTLS handshake using its new IP address. This means that the DTLS channel established during network access is transient and it is closed by the relay device once the handshake is finished.

4.3 Key Management

This section describes the details of key management for unicast and group communication. The goal of this phase is to set up an AMIKEY crypto-session bundle (CSB) for unicast as well as group communication. A CSB is built from some root keying material (the TEK Generation Key (TGK)) and some random bits ("RAND"). AMIKEY then defines a lightweight mechanism for the derivation and management of fresh purpose-specific keys, called Traffic Encryption Keys (TEKs), that are used to secure the communication links. We now explain, for both the unicast and multicast case, how the root keying material, i.e., the TGK, is obtained, how the CSB is negotiated and set up, and finally how TEKs are requested and generated.

4.3.1 Management of unicast keys

DTLS runs on the transport layer, and thus we can use it directly to protect the applications. Two communicating devices can establish a pairwise keys using a polynomial scheme, in which each device's polynomial share is used to facilitate fast pairwise key agreement between them. This pairwise key serves as the PSK in DTLS-PSK [RFC4279] enabling any two applications running on the devices to derive a session key. In detail:

1. Two applications running on the devices "D1" and "D2" start a DTLS-PSK handshake. They exchange their identities ID_D1 and ID_D2 as extensions to the first two handshake messages, the "ClientHello" and "ServerHello".
2. Both devices then generate a pairwise key, e.g., if a polynomial scheme is used, they use their polynomial shares and their respective identities to arrive on a pairwise key: K(D1,D2) = F(ID_D1, ID_D2) = F(ID_D2, ID_D1).
3. The derived key K(D1,D2) is used as the PSK to complete the DTLS-PSK handshake. This PSK can be regarded as the TGK.
4. The result of the DTLS-PSK handshake is a session key used to protect the communication link between the two applications on

both devices.

4.3.2 Management of multicast keys

```
Joining Device                                   Domain Manager

ClientHello + extension      -------->
                             <--------  ServerHello + Extension

<----------Continue with the rest of DTLS Handshake----------->

                                                Generate AMIKEY
                                                parameters

                                          DTLS(CSB_ID, TGK, MIC)
                             <--------    protected with DTLS

DTLS(ACK, MIC)
protected with DTLS          -------->

                                    DTLS(CSB_ID,KEY_ID,RAND,MIC)
                             <--------    protected with DTLS

DTLS(ACK, MIC)
protected with DTLS          -------->
```

Figure 3: DTLS-based Multicast Key Management

The joining device first indicates during the network access phase that it wishes to join a certain multicast group by adding a request with the group id in the extension part of the "ClientHello". The domain manager then issues the multicast group keys to the device, if it has been authorized to join. It is carried as payload over DTLS record layer after the initial handshake has finished as shown in Figure 3. Two new "Content Types" for the DTLS header have been defined for this purpose to distinguish between the DTLS protected application data and key management data. They are the necessary parameters to set up a CSB, i.e. (1) "Master Key Data" (e.g., the TGK) and (2) "Security Parameters Data" (e.g., the "RAND" values). The fresh TEKs can then be derived from the CSB by every group member for secure group communication.

When a device leaves a group, the domain manager deletes it from the list of authenticated nodes, increases the TEK_ID and starts a new TEK derivation process. The parameters required for generation of the fresh TEK are encrypted so that the leaving device cannot derive the new TEK and cannot rejoin without re-authorization.

4.4 Secure Uni- and Multicast Communication

Once the pairwise session keys or multicast group session key has been derived, a secure channel can be created to transport data (CoAP messages) between the devices in the IoT network. For this, we rely on the DTLS record-layer to create a secure transport layer for CoAP. In this case, the standard DTLS is used to secure the unicast communication.

For multicast communication, the combination of our proposed handshake protocols with the usage of DTLS record-layer for multicast security is based on [I-D.keoh-multicast-security].

4.4.1 Unicast Communication

Any pair of devices in the IoT domain that wish to communicate with each other, establish a CSB and derive a fresh unicast TEK through DTLS as described in Section 4.3.1. The TEK is used in the DTLS record layer (based on AES-CCM) to protect the message exchange between two applications. AES-CCM [RFC3610] is an AES mode of operation that defines the use of AES-CBC for MAC generation with AES-CTR for encryption. The CCM counter (corresponding to the DTLS epoch and sequence number fields) are initialized to 0 upon TEK establishment and used in the nonce construction in a standard way.

4.4.2 Multicast Communication

The multicast solution relies on IP multicast (i.e., an IP multicast address) for routing purposes and adds a security layer on top. To protect the communication, a group of devices establishes a multicast CSB and fresh TEKs using DTLS as described in Section 4.3.2. The TEK is then used to protect CoAP messages transported over the DTLS record layer in AES-CCM mode and routed via IP-multicast. Each device in the group uses a CCM nonce composed of a fixed common part (the content type from the DTLS record layer and the group identifier) and a variable part (the epoch and sequence number fields in the DTLS record layer). This ensures a unique nonce for each message [RFC3610] in the context of a same key.

5 Implementation and Evaluation

This section describes the prototype of our security solution and evaluates the memory and communication overheads.

5.1 Prototype Implementation

The prototype is written in C and run as an application on Contiki OS 2.5 [Dunkels-contiki], an event-driven open source operating system

for constrained devices. They were tested in the Cooja simulator and then ported to run on Redbee Econotag hardware, which features a 32-bit CPU, 128 KB of ROM, 128 KB of RAM, and an IEEE 802.15.4 enabled-radio with an AES hardware coprocessor. The prototype comprises all necessary functionalities to adapt to the roles as a domain manager or a joining device.

The prototype is based on the "TinyDTLS" [Bergmann-Tinydtls] library and includes most of the extensions defined in Section 4 and the adaptation as follows:

(1) We disabled the cookie mechanism in order to fit messages to the available packet sizes and hence reducing the total number of messages when performing the DTLS handshake.

(2) We used separate delivery instead of flight grouping of messages and redesigned the retransmission mechanism accordingly.

(3) We modified the "TinyDTLS" AES-CCM module to use the AES hardware coprocessor.

(4) The Relay Element functionality for multi-hop scenario has not been implemented.

(5) We expanded the DTLS state machine with the necessary additions for our key management solution.

The following subsections further analyze the memory and communication overhead of the solution in a single-hop scenario.

5.2 Memory Consumption

Table 1 presents the codesize and memory consumption of the prototype differentiating (i) the state machine for the handshake, (ii) the cryptographic primitives, (iii) the key management functionality and (iv) the DTLS record layer mechanism for secure multicast communications.

The use of DTLS appears to incur large memory footprint both in ROM and RAM for two reasons. First, DTLS handshake defines many message types and this adds more complexity to its corresponding state machine. The logic for message re-ordering and retransmission also contribute to the complexity of the DTLS state machine. Second, DTLS uses SHA2-based crypto suites which is not available from the hardware crypto co-processor.

	DTLS	
	ROM	RAM
State Machine	8.15	1.9
Cryptography	3.3	1.5
Key Management	1.0	0.0
DTLS Record Layer	3.7	0.5
TOTAL	16.15	3.9

Table 1: Memory Requirements in KB

5.3 Communication Overhead

We evaluated the communication overhead in the context of "network access" and multicast "key management". In particular, we examine the message overhead and the number of exchanged bytes under ideal condition without any packet loss in a single-hop scenario, i.e., domain manager and a joining device are in communication range.

	DTLS
No. of Message	12
No. of round trips	4
802.15.4 headers	168B
6LowPAN headers	480B
UDP headers	96B
Application	487B
TOTAL	1231B

Table 2: Communication overhead for Network
Access and Multicast Key Management

Table 2 summarizes the required number of round trips, number of messages and the total exchanged bytes for the DTLS-based handshake carried out in ideal conditions, i.e., in a network without packet losses. DTLS handshake is considered complex as it involves the exchange of 12 messages to complete the handshake. Further, DTLS runs on top the transport layer, i.e., UDP, and hence this directly increases the overhead due to lower layer per-packet protocol headers.

5.4 Message Delay, Success Rate and Bandwidth

Section 5.3 provided an evaluation of the protocol in an ideal condition, thus establishing the the baseline protocol overhead. We further examined and simulated the protocol behavior by tuning the packet loss ratio. In particular, we examined the impact of packet loss on message delay, success rate and number of messages exchanged in the handshake.

We consider a complete handshake to include the protocols to perform "network access" and "multicast key management". Figure 4 shows the percentage of successful handshakes as a function of timeouts and packet loss ratios. As expected, a higher packet loss ratio and smaller timeout (15s timeout) result in a failure probability of completing the DTLS handshake. When the packet loss ratio reaches 0.5, practically no DTLS handshake would be successful.

```
      100 |+
P         | +
E     80  |  ++
R         |    ++
C     60  |      +
E         |       +
N     40  |        +
T         |         ++
A     20  |            +
G         |             +++++
E      0 +------------------+++++++++-->
        0 0.1 0.2 0.3 0.4 0.5

         packet loss ratio (15s timeout)
```

Figure 4: Average % of successful handshakes

Delays in network access and communication are intolerable since they lead to higher resource consumption. As the solution relies on PSK, the handshake protocol only incurs a short delay of a few milliseconds when there is no packet loss. However, as the packet loss ratio increases, the delay in completing the handshake becomes significant because loss packets must be retransmitted. Our implementation shows that with a packet loss ratio of 0.5, the the times to perform network access and multicast key management could take up to 24s.

Finally, another important criterion is the number of messages exchanged in the presence of packet loss. A successful handshake could incur up to 35 or more messages to be transmitted when the packet loss ratio reaches 0.5. This is mainly because the DTLS retransmission is complex and often requires re-sending multiple

messages even when only a single message has been lost.

6. Conclusions and future work

This Internet Draft presented an approach to secure the IP-based Internet of Things using DTLS with the focus on (i) secure network access, (ii) key management, and (iii) secure uni- and multicast communication. Apart from secure unicast communication with DTLS, there are no standard IP solutions in IETF for these unavoidable problems when deploying an IoT network. We have shown that the existing IP-based security protocol, i.e., DTLS can be used and adapted to cater for low resource devices (bandwidth, memory and CPU) and new communication patterns such as multicast and multi-hop network access.

As a proof of concept, we implemented the an architecture based on DTLS over Contiki OS running on a Redbee Econotag. Our work has shown that the re-use of the existing standardized DTLS protocol to solve these problems with a compromise in efficiency is feasible. We hope that our work provides valuable protocol designs and evaluation results (with their pros and cons) which can provide the much needed direction for the standardization effort in IETF to ensure best solutions are adopted.

7 Security Considerations

This document discusses various design aspects for of DTLS implementations. As such this document, in entirety, concerns security.

8 IANA Considerations

tbd

9. Acknowledgements

The authors greatly acknowledge discussion, comments and feedback from Dee Denteneer and Jan Henrik Ziegeldorf. We also appreciate the prototyping and implementation efforts by Pedro Moreno-Sanchez and Francisco Vidal-Meca who worked as interns at Philips Research.

10 References

10.1 Normative References

[RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer

Security", RFC 4347, April 2006.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

[RFC4279] Eronen, P., Ed., and H. Tschofenig, Ed., "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.

[RFC3610] Whiting, D., Housley, R., and N. Ferguson, "Counter with CBC-MAC (CCM)", RFC 3610, September 2003.

9.2 Informative References

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-12 (work in progress), October 2012.

[I-D.alexander-roll-mikey]
Alexander, R., and Tsao, T. "Adapted Multimedia Internet KEYing (AMIKEY): An extension of Multimedia Internet KEYing (MIKEY) Methods for Generic LLN Environments", draft-alexander-roll-mikey-lln-key-mgmt-04 (work-in-progress), September 2012.

[Blundo-polynomial]
Blundo, C., De Santis, A., Herzberg, A., Kutten, S., Vaccaro, U., and Yung, M. "Perfectly-Secure Key Distribution for Dynamic Conferences", Advances in Cryptology (CRYPTO'92), 1993.

[RFC6345] Duffy, P., Chakrabarti, S., Cragie, R., Ohba, Y., and Yegin, A. "Protocol for Carrying Authentication for Network Access (PANA) Relay Element", RFC 6345, August 2011.

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., and Bormann, C. "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

[I-D.keoh-multicast-security]
Keoh, S., Garcia-Morchon, O., and Kumar, S. "DTLS-based Multicast Security for Low-Power and Lossy Networks (LLNs)" (work-in-progress), October 2012.

[Dunkels-Contiki]
Dunkels, A., Gronvall, B., and Voigt, T. "Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors", In Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, IEEE, 2004.

[Bergmann-Tinydtls] Bergmann, O. "TinyDTLS - A Basic DTLS Server Template", http://tinydtls.sourceforge.net, 2012.

Authors' Addresses

Sye Loong Keoh
University of Glasgow
Republic PolyTechnic, 9 Woodlands Ave 9
Singapore 838964
SG

Email: SyeLoong.Keoh@glasgow.ac.uk

Sandeep S. Kumar
Philips Research
High Tech Campus 34
Eindhoven 5656 AE
NL

Email: sandeep.kumar@philips.com

Oscar Garcia-Morchon
Philips Research
High Tech Campus 34
Eindhoven 5656 AE
NL

Email: oscar.garcia@philips.com