Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 10, 2015

A. He, Ed.
Huawei
B. Sarikaya, Ed.
Huawei USA
March 9, 2015

IoT Security Bootstrapping: Survey and Design Considerations
draft-he-6lo-analysis-iot-sbootstrapping-00

Abstract

This document presents the importance of security bootstrapping for IoT networks, analyzes the state-of-the-art works in standard organizations and discusses what should be considered when designing the secure bootstrapping mechanism.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 10, 2015.

Copyright Notice

Table of Contents

1. Introduction

An Internet of Things (IoT) network is composed of connected things that cooperate together to accomplish tasks such as smart buildings, smart environment monitoring system, intelligent transport system, etc. The size of IoT varies from tens to thousands depending on the application, and things in an IoT network might be produced by different vendors and they are normally heterogeneous with various constraints e.g. power supply, communication capability, CPU and memory.

IEEE 802.15.4 specifies the physical layer and media access control for low-rate wireless personal area networks (LR-WPANs). It is widely used in wireless sensor networks nowadays and is foreseen as the most used lower layer protocol for low rate IoT networks with resource constrained devices. In IETF, 6LoWPAN (concluded) developed RFC 4944 [RFC4944] to describe how to transmit IPv6 packets over 802.15.4, and support mesh routing in LR-WPANs. 6lo defines generic IPv6 packet header compression method [RFC7400] for LR-WPANs. 6tisch

tries to build adaptation protocols for IEEE 802.15.4e specification. Roll develops routing protocol RPL [RFC6550] for IPv6 based low power and lossy networks. Note that IEEE 802.15.4 can be applied to mobile nodes, routing protocols such as AODV [RFC3561], DSL [RFC4728], OLSR [RFC3626] by MANET group are also widely used. CoAP [RFC7252] from

CoRE defines a UDP based web transfer protocol for machine- to-machine (M2M) applications such as smart energy and building automation.

The above mentioned protocols provide different selections of IoT protocol stacks to fulfill specific tasks based on IEEE 802.15.4. At the start-up phase of a network or after the provisioned communications have failed, bootstrapping is typically required to configure nodes at all layers, including anything from link-layer information (i.e., wireless channels, link-layer encryption keys) to application-layer information (i.e., network names, application encryption keys). It can be realized either manually via user interface or automatically via interaction between nodes.

Traditional bootstrapping approaches tend to impose configuration burdens upon users. For example, users need to follow a series of instruction steps for configuration. Configuring IoT devices becomes more complicated since they don't always provide user interface to input all necessary information, and the scale of the IoT network can be large, dynamic or error prone. As a result, human intervention is expensive and not efficient in those situations. This motivates the need for self-organization and automatic self-bootstrapping in IoT. Enabling a plug & play framework not only reduces human efforts in configuring IoT but also improve the scalability and flexibility. This draft presents a survey of the state-of-the-art works on bootstrapping/networking in IETF, ZigBee Alliance, IEEE and Thread group, and the design considerations for security bootstrapping are derived.

2. Terminology

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

3. Analysis of Related State-of-the-art Works

Bootstrapping is required at all layers, where different conditions and information should be transferred for different protocols. This section provides analysis on the existing bootstrapping works in standard organizations and summarizes the concerns.

3.1. Security bootstrapping

Security bootstrapping includes the authentication of devices to establish trust relationships in a network, as well as transferring security parameters and keying materials. Security bootstrapping is

believed as the fundamental part of bootstrapping, because once secure and authentic channels are established, the bootstrapping of all other information can be conducted as ordinary secured communications. Accordingly, many works focus on security bootstrapping and device authentication. In IETF, [I-D.pritikin-anima-bootstrapping-keyinfra] is proposed in Anima, [I-D.sarikaya-6lo-bootstrapping-solution] is proposed in 6lo, [I-D.struik-6tisch-security-considerations] is in 6tisch, [I-D.kwatsen-netconf-zerotouch] is in Netconf, and [I-D.he-iot-security-bootstrapping] is proposed for bootstrapping IEEE 802.15.4 based IoT networks. ZigBee IP stack is developed by ZigBee Alliance and it supports EAP-TLS and PANA as authentication protocols. In Thread Group, a networking solution is developed. The devices are authenticated through pre-installed codes. IEEE 802.15.4 also defines two-step mechanism for nodes joining network with layer 2 authentication without considering IP infrastructure.

3.1.1. Authentication framework

The arguments on authentication framework focus on EAP, PANA, HIP-DEX, 802.1X via EAPOL, and IKEv2.

[I-D.oflynn-core-bootstrapping] relates the aforementioned authentication frameworks into IEEE 802.15.4 and requirements in order to use them for bootstrapping procedure.

- If PANA is used, a new entity called PANA Relay Element should be added in the architecture and behavior of PANA RE needs to be defined [RFC6345]; New AVPs needed for PANA Relay Element

operation for relaying messages from the client to the authenticator and vice versa are required to be specified. If PANA is used to securely distribute group key [RFC6786] from the PANA Authentication Agent to the PANA Client using AES Key Wrap with padding algorithm, an extension to PANA needs to be defined.

- If HIP-DEX is used, the initiator should be able to get the IP address of the responder, either using DNS infrastructure or local configuration.

- If 802.1X is used, a special value in the Frame Type subfield of the Frame Control Field of IEEE 802.15.4 MAC header should be assigned to indicate the type of the payload. Group addresses for 802.15.4 corresponding to EAPOL Group Address Assignments defined in Table 11.1 of [IEEE802.1x] are required, especially for EAPOL-Start packet. The mapping of MAC frames and security level to different types should be defined, for instance: which MAC frames of beacon, data, acknowledgment and MAC command as defined in [IEEE802.15.4] with what security levels are mapped to controlled

port, which MAC frames with what security levels are mapped to uncontrolled port and which MAC frames are never mapped to any of controlled/uncontrolled port (i.e., the payload of those frames are used by the MAC-layer itself and never used by upper layers).

[I-D.garcia-core-security] discusses about using Internet Key Exchange protocol version 2 (IKEv2) as authentication method. It summarizes that IKEv2 can perform key exchanges and the setup of security associations without online connections to a trust center. It provides end-to-end security, and supports host mobility with MOBIKE extension. However, MOBIKE mandates the use of IPsec tunnel mode which requires to transmit an additional IP header in each packet. This additional overhead could be alleviated by using header compression methods or the Bound End-to-End Tunnel (BEET) mode [I-D.nikander-esp-beet-mode], a hybrid of tunnel and transport mode with smaller packet headers.

Several EAP methods have been standardized for different purposes. One widely used method is the EAP-TLS [RFC7250] which enables mutual authentication and distribute keying material to secure subsequent communications. However it only supports certificate-based mutual authentication, thus public key infrastructure is required and

fragmentation is needed when using IEEE 802.15.4 to exchange authentication messages.

ZigBee Alliance specified an IPv6 stack aimed at IEEE 802.15.4 devices mainly used in smart meters developed primarily for SEP 2.0 (Smart Energy Profile) application layer traffic [SEP2.0]. This specification assumes Class 2 devices which have 50 KiB of RAM and 250 KiB of flash memory [RFC7228]. Some devices in such systems have more resources and processing power (e.g. ARM9 core, MiBs RAM/ Flash). For security bootstrapping, ZigBee IP uses EAP-TLS.

Authentication that is not based on certificates reduces cost of certificate management and fewer messages are needed to be exchanged between client and server. [I-D.sarikaya-6lo-bootstrapping-solution] proposes to use raw public keys via EAP-TLS, thus extension to EAP-TLS is indicated. Note that EAP requires exchanging the device identity in plain text at the beginning, but how to protect the privacy information indicated in the device ID is out of concern of EAP methods.

EAP-PSK [RFC4764] is another EAP method. It realizes mutual authentication and session key derivation using a Pre-Shared Key (PSK). Normally four messages are exchanged in the authentication process. Once the authentication is successful, EAP-PSK provides a protected communication channel.

EAP-IKEv2 [RFC5106] is an EAP method based on IKEv2.. It provides mutual authentication and session key establishment between an EAP peer and an EAP server. It supports authentication techniques that are based on different credentials including asymmetric key pairs, symmetric keys and passwords. Besides, it is possible to use a different authentication credential in each direction. For example, the EAP server authenticates itself using public/private key pair and the EAP peer using symmetric key. As a result different combinations of credentials are expected to be used in practice. Compared with EAP-TLS and EAP-PSK, EAP-IKEv2 supports mobility and different authentication techniques.

[I-D.kumar-6lo-selective-bootstrap] presents a selective bootstrapping/commissioning method by introducing the concept of Commissioning Tool (CT). In this method the devices are let to

connect to the network and execute 6LowPAN neighbor discovery protocol and have an IPv6 address before they are authenticated. Then the devices are selected one by one in some order to communicate with the CT via untrusted constructed route. Once the ID of joining device is authenticated, CT sends the layer-2 key material to the device via secured channel, which is established by DTLS by exchanging credential material installed during manufacturing.

The bootstrapping method in [I-D.kumar-6lo-selective-bootstrap] creates security risks for the network by

1. letting the devices have IP addresses for layer3 communication before authentication.

2. constructing routing topology before devices are authenticated.

3. establishing transport layer security before layer-2 security.

However, such a protocol could be justified in some application domains like lightning control systems.

There is work going on in the IEEE 802.15.9 task group which specifies a way to transport existing key management protocols (KMP) over the 802.15.4 frames. The new feature would allow running IKEv2, EAP,PANA, 802.1X, HIP and Dragonfly over the IEEE 802.15.4 and generate keys for 802.15.4 security and protect all messages between the two nodes [IEEE802.15.9]. It would be desired if the security bootstrapping procedure reuses the KMPs that supported by lower layers to reduce cost.

Table 1 summarizes the authentication frameworks and credential materials of the aforementioned solutions.

Referenced solution	Authentication method	Credential material
[I-D.pritikin-anima-bootstrapping-keyinfra]	802.1x-EAPOL, EAP-TLS, EAP-IKEv2	802.1AR certificate
[I-D.sarikaya-6lo-bootstrapping-sol	EAP-TLS	Raw public

ution]	(modified)	key
[I-D.struik-6tisch-security-conside rations]	Joining Protocol (undefined)	Certificat e
[I-D.kwatsen-netconf-zerotouch]	Unspecified (EAP-TLS might be used)	X.509 cert ificate
[I-D.he-iot-security-bootstrapping]	EAP, PANA	Unspecifie d
[I-D.kumar-6lo-selective-bootstrap]	Selected by Commissioner with CT	PSK defined in CT
ZigBee IP stack based Smart Energy	EAP-TLS, PANA	Certificat e
Thread networking	Unknown	Product install codes

Table 1

3.1.2. Credential Material and Architecture

The trust relationship can be established by exchanging credential materials, which can be asymmetric with user authentication or with certificate authority, or symmetric pre-shared key configured by network developer. In certificate authority (CA), a typical public key infrastructure (PKI) is used, meaning that a set of hardware, software, people, policies, and procedures are needed to create, manage, distribute, use, store, and revoke digital certificates. The public keys are obtained in PKI containers, and both ends are validated using trust anchors based on a certification authority (CA). [I-D.pritikin-anima-bootstrapping-keyinfra] uses 802.1AR certificate, [I-D.kwatsen-netconf-zerotouch] uses X.509 certificate. Certificate mechanism provides high security however it can add a complicated trust relationship that is difficult to validate. When it comes to large scale IoT networks, certificate management and distribution will raise scalability and flexibility issue. Besides, the time spent and CPU occupied by the cryptographic operations is non-trivial when this mechanism is implemented on computational

constraint devices. Since some IEEE802.15.4 technologies including

802.15.4e only allows 127 Octets maximum payload, fragmentation is unavoidable, which indicates that a large amount of data is transmitted and communication overhead is heavy. The public-key based handshake process of EAP-TLS is part of the bottleneck that significantly degrades the performance. Designers are forced to use highly efficient protocols for the sake of ensuring the computational complexity of security algorithms as low as possible.

In today's IoT, most common architectures are fully centralized in a sense that all the security relationships within a segment are handled by a central party.

The 802.1x framework, the architecture proposed in [I-D.pritikin-anima-bootstrapping-keyinfra] and the ZigBee IP smart energy solution are centralized. A centralized authentication architecture allows for central management of devices and keying materials as well as for the backup of cryptographic keys. As a result there is no high requirement on network devices in a centralized architecture. However it also represents a single point of failure and is more suitable for static network where the route to the trust center/AAA server is stable.

The self-signed certificates are commonly used in smaller deployments where they are distributed to all involved protocol endpoints out-of-band, thus CA and certificate management are not required. This practice does, however, still require the overhead of the certificate generation even though none of the information found in the certificate is actually used.

The raw public key method is proposed to generate light weight certificate, which can significantly reduce overhead. However, the self-signed certificate and raw public key only prove the possession of the private-public key pair and are unable to prove whether the owner is legitimate.

The pre-shared key based mechanisms are more suitable for constrained environments, e.g. wireless communications, and limited CPU power devices. It enables mutual authentication, meanwhile requires less cryptographic operations and less communication overhead compared with certificate based mechanism. However, traditional approaches of key generation/distribution tend to impose configuration burdens upon users. For example, users need to follow a series of instruction steps for WiFi Protected Access 2, Pre-shared key (WPA2-PSK) configuration, even though the pre-shared key mode is the simplest option for using WPA. Establishing security among IoT devices becomes more complicated since they don't always provide user interface to input necessary security information.

As discussed, the authentication of self-signed certificate and pre-shared key mechanisms are distributed. Distributed architecture allows creating ad-hoc security domains that might not require a single online management entity and are operative in a much more stand-alone manner. In this case, hardware should be configured to be able to authenticate and verify other peers.

In today's IoT, most common architectures are fully centralized in a sense that all the security relationships within a segment are handled by a central party.

The Thread protocol is expected to use product install codes as authentication material. Currently not enough details are available on the Thread protocol.

Physical unclonable function (PUF) arises as a promising authentication technology. PUF is a physical entity that is embodied in a physical structure and is easy to evaluate but hard to predict. Further, an individual PUF device must be easy to make but practically impossible to duplicate, even given the exact manufacturing process that produced it. In this respect it is the hardware analog of a one-way function. PUFs can serve as a root of the trust and can provide a key which cannot be easily reverse engineered. Temperature and aging have been given special attention on developing reliable PUF [MIT2014].

3.2. Higher Layer Protocol Use After/During Bootstrapping

Configurations of parameters for other protocols are important as well to ensure a successful networking. Those parameters are transferred upon a successful security bootstrapping.

The IP address configuration is a major issue which must be solved before any other higher layer service can start. It can be locally pre-configured, auto-configured or managed from a third party tool.

- Pre-configured: is mainly what is done today. No further network service is needed, the assignment is done from a planning/ commissioning tool instead. This method requires human interaction, devices with IP configured are trusted by default. scalability and flexibility cannot be satisfied in this case.

- Auto-configuration: the device creates its IP address itself, applying one of the algorithms specified in the relevant standards, e.g. ZigBee IP solved this problem by using SLAAC IPv6 addresses based on the EUI-64;

[I-D.pritikin-anima-bootstrapping-keyinfra] suggests to obtain an IP address using existing methods, such as SLAAC or DHCPv6. RPL

[RFC6550] is a special routing protocol that generates for each device an IP prefix based on the constructed routing topology, thus special attention should be paid as chicken/egg issue arises when relay of authentication is needed by the network level bootstrapping. The auto-configured IP address may need to perform a check for duplicates (i.e. APIPA17). Encoding of semantics into the address may need information from lower layer (see above) or from network service. Note, this only works for so-called link local-addresses which are valid only in one Ethernet domain.

- Managed: pre-planned addresses are assigned by means of a third party database, such as DHCP, a central server.

4. Role of IoT Security Bootstrapping

Figure 1 shows a network life cycle: after IoT devices being deployed in field, the security bootstrapping starts. Devices are authenticated, keying materials are exchanged for securing subsequent configuration/data exchange messages. The device gets an IP address and joins the network.

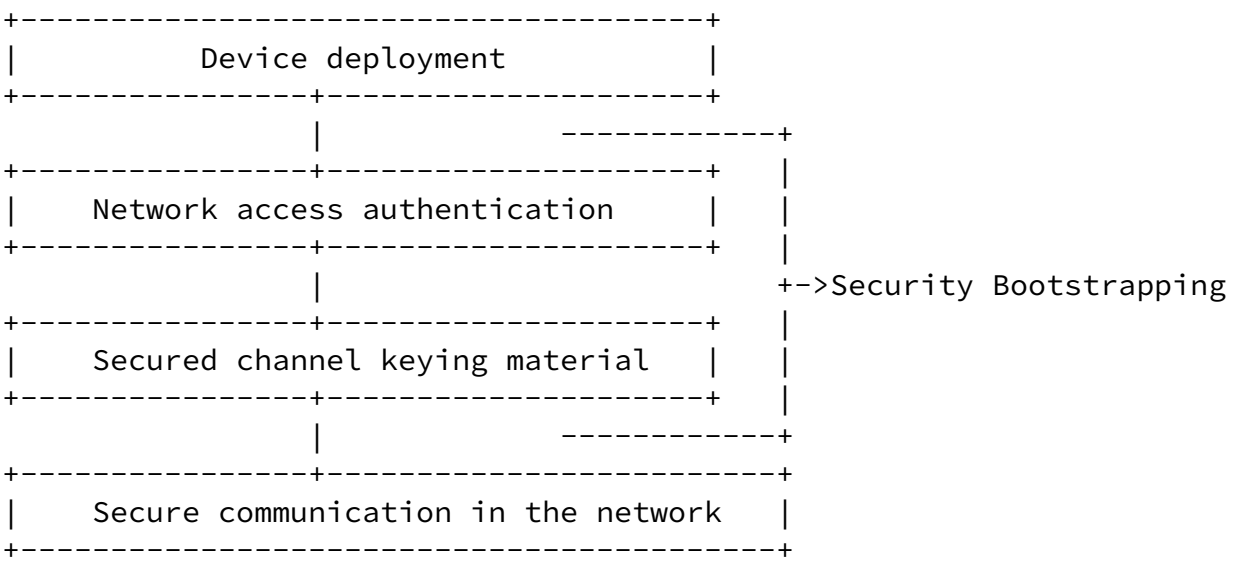

Figure 1: IoT Life Cycle

5. Design Considerations

IoT can be deployed in different environments for different applications, which calls for protocols with options where a set of options is selected to construct a protocol stack that fits for a given environment, e.g. home, enterprise or industrial. The deployment and configurations can also be divided into two types, one is for static network, and the other is for dynamic network.

IoT developed in buildings, homes, or industrial areas are often static. A general approach is that a network engineer plans the locations for each device and determines topology of network based on deployment environment and channel estimation. Then the key devices (e.g. sink nodes, or parent nodes of a routing protocol) are installed before deploying other devices. Upon successful installation, the device is plugged and security bootstrapping is run in either centralized or distributed manner with pre-configured credential material. The device is at work after all the protocols are successfully bootstrapped. When a new device joins an existing network, the joining device bootstrapping procedure is triggered by itself.

In a dynamic network where devices come and go, their IPv6 addresses might also change. Bootstrapping/re-commissioning at network level is more frequently required than that in static network, hence minimum human interaction is highly preferred. Reducing communication overhead will improve the efficiency of networking, and this is especially useful for low bandwidth and low rate IEEE 802.15.4.

Mains-powered devices can stay continuously connected to the network. Normally-off power strategy can be used for battery powered devices where the devices sleep long periods of time and stay disconnected and reattach to the network after it is woken up. Between these two extremes, there is low-power device mode where the devices need to be able to communicate on a relatively frequent basis [RFC7228].Bootstrapping protocol needs to be able to take into consideration these power levels in the design.

The order of bootstrapping is another concern in designing the bootstrapping protocol. The devices could arbitrarily be

bootstrapped as they join the network, especially in dynamic topologies. In static topologies the order could be completely installation and installer dependent and could be optimized to lower cost and could be independent of network topology [I-D.kumar-6lo-selective-bootstrap]. The order is also dependent on the architecture of authentication. For centralized architecture, incremental approach is recommended by [I-D.he-iot-security-bootstrapping] , [I-D.garcia-core-security] and [I-D.sarikaya-6lo-bootstrapping-solution], whereas a selective order can be specified by CT [I-D.kumar-6lo-selective-bootstrap] and special attention should be paid on the secured channel establishment via untrusted route. For decentralized architecture, the mutual-authentication is realized between equal peers in pure mesh topology without any preferred order and network keys can be distributed by cluster heads once clusters are formed.

Some mandatory considerations can be derived from different applications for IoT security bootstrapping mechanism:

5.1. Able to clearly define security dependency and trust domains

Things of IoT are more related to private data, thus trust increases its importance. It is easy to introduce a new node in a deployed IoT to capture and analyze the data traffic. As a result,

a. Security dependencies between different devices must be clarified. Circular dependencies must be avoided.

b. The designed protocol should enable mutual authentication between devices running the security bootstrapping protocol. Proper authentication material and mechanism should be chosen.

c. The security bootstrapping protocol processing devices should agree upon the security associations (e.g. key materials, algorithms etc.) for securing their communications before exchanging any protocol packets.

5.2. Cross-layer design

The security bootstrapping method should take into account the features and requirements of full stack protocols that are selected

for an IoT network. Security bootstrapping in collaboration with other networking protocols is likely to produce a comprehensive solution.

Cooperative communication and scheduling among neighboring things at lower layer will reduce the possibility of network congestion and assist finishing bootstrapping efficiently. Different power modes should be considered by the designed protocol.

As discussed in Section 3.2, higher layer protocols impact the procedure of bootstrapping. During network start-up, link local IP address should be assigned in order to run PANA/TLS to forward authentication messages by IoT routing protocols such as AODV and DSR in MANET. However, the RPL for LLN configures IP addresses for all the devices during/ at the end of routing procedure, which may create a chicken/egg issue when PANA/TLS are also used. 802.1X uses link layer address so no IP address is needed.

5.3. Reduce human interaction to the minimum

Configuring IoT devices can be complicated since they don't always provide user interface to input all necessary information, and the scale of the IoT network can be large, dynamic or error prone.

Besides, IoT network users usually do not have expertise in networking, this motivates self-organizing IoT network protocol that start from security bootstrapping. As a result, the design of bootstrapping protocol should be able to reduce human interaction to the minimum.

5.4. Able to resist attacks

The designed bootstrapping protocol should be able to resist attacks and protect CIA triad. Typical threat modeling approaches (e.g. STRIDE) should be used to guide the design of bootstrapping architecture and procedure. STRIDE categorizes attack into spoofing, Tampering with data, Repudiation, Information disclosure, Denial of service and Elevation of privilege.

5.5. Low computation cost and communication overhead

The amount of transmitted data and the complexity of data processing

should be optimized to the minimum to save computation and communication cost.

6. Security Considerations

TBD

7. Acknowledgements

TBD

8. References

8.1. Normative References

[IEEE802.15.4]
IEEE Standard, , "IEEE Std. 802.15.4-2011", October 2011, <http://standards.ieee.org/findstds/standard/802.15.4-2011.html>.

[IEEE802.15.9]
IEEE P802.15.9/D01, "IEEE Draft Recommended Practice for transport of key management protocol (KMP) datagrams", November 2014, <http://grouper.ieee.org/groups/802/15/private/members_area.html>.

[IEEE802.1x]
IEEE Std 802.1X-2010, "IEEE 802.1X Port-Based Network Access Control", February 2010, <http://standards.ieee.org/getieee802/download/802.1X-2010.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3561] Perkins, C., Belding-Royer, E., and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", RFC 3561, July

2003.

[RFC3626] Clausen, T. and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", RFC 3626, October 2003.

[RFC4728] Johnson, D., Hu, Y., and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", RFC 4728, February 2007.

[RFC4764] Bersani, F. and H. Tschofenig, "The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method", RFC 4764, January 2007.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, August 2007.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC5106] Tschofenig, H., Kroeselberg, D., Pashalidis, A., Ohba, Y., and F. Bersani, "The Extensible Authentication Protocol-Internet Key Exchange Protocol version 2 (EAP-IKEv2) Method", RFC 5106, February 2008.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[RFC5487] Badra, M., "Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode", RFC 5487, March 2009.

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

[RFC6550] Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, March 2012.

[RFC6786] Yegin, A. and R. Cragie, "Encrypting the Protocol for Carrying Authentication for Network Access (PANA) Attribute-Value Pairs", RFC 6786, November 2012.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.

[RFC7250] Wouters, P., Tschofenig, H., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, June 2014.

[RFC7251] McGrew, D., Bailey, D., Campagna, M., and R. Dugal, "AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS", RFC 7251, June 2014.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

[RFC7400] Bormann, C., "6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 7400, November 2014.

[SEP2.0] ZigBee Alliance, "ZigBee IP Specification", March 2014, <hhttp://www.zigbee.org/non-menu-pages/zigbee-ip-download/>.

8.2. Informative References

[I-D.garcia-core-security]
Garcia-Morchon, O., Kumar, S., Keoh, S., Hummen, R., and R. Struik, "Security Considerations in the IP-based Internet of Things", draft-garcia-core-security-06 (work in progress), September 2013.

[I-D.he-iot-security-bootstrapping]
ana.hedanping@huawei.com, a., "Security Bootstrapping of IEEE 802.15.4 based Internet of Things", draft-he-iot-security-bootstrapping-00 (work in progress), January 2015.

[I-D.kumar-6lo-selective-bootstrap]
Kumar, S. and P. Stok, "Security Bootstrapping over IEEE 802.15.4 in selective order", draft-kumar-6lo-selective-bootstrap-00 (work in progress), March 2015.

[I-D.kwatsen-netconf-zerotouch]
Watsen, K., Hanna, S., Clarke, J., and M. Abrahamsson, "Zero Touch Provisioning for NETCONF Call Home (ZeroTouch)", draft-kwatsen-netconf-zerotouch-01 (work in progress), February 2014.

[I-D.nikander-esp-beet-mode]
Nikander, P. and J. Melen, "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-09 (work in progress), August 2008.

[I-D.oflynn-core-bootstrapping]
Sarikaya, B., Ohba, Y., Cao, Z., and R. Cragie, "Security Bootstrapping of Resource-Constrained Devices", draft-oflynn-core-bootstrapping-03 (work in progress), November 2010.

[I-D.pritikin-anima-bootstrapping-keyinfra]
Pritikin, M., Behringer, M., and S. Bjarnason, "Bootstrapping Key Infrastructures", draft-pritikin-anima-bootstrapping-keyinfra-01 (work in progress), February 2015.

[I-D.sarikaya-6lo-bootstrapping-solution]
Sarikaya, B., "Secure Bootstrapping Solution for Resource-Constrained Devices", draft-sarikaya-6lo-bootstrapping-solution-00 (work in progress), June 2013.

[I-D.struik-6tisch-security-considerations]
Struik, R., "6TiSCH Security Architectural Considerations", draft-struik-6tisch-security-considerations-01 (work in progress), January 2015.

[MIT2014] Herder, C., Farinaz Koushanfar, F., and S. Srinivas Devadas, "Physical Unclonable Functions and Applications: A Tutorial", Proceedings of the IEEE , vol. 102, no. 8,

pp. 1126-1141, August 2014.

Authors' Addresses

Ana(Danping) He (editor)
Huawei
Building Q14, 156 Beiqing Road
Beijing 100095
China

Email: ana.hedanping@huawei.com

Behcet Sarikaya (editor)
Huawei USA
5340 Legacy Dr. Building 3
Plano, TX 75024

Email: sarikaya@ieee.org