6tisch Working Group
Internet-Draft
Intended status: Informational
Expires: April 23, 2017

M. Richardson
Sandelman Software Works

October 20, 2016

6tisch Secure Join protocol

draft-richardson-6tisch-dtsecurity-secure-join-01

Abstract

Charter: The WG will continue working on securing the join process and making that fit within the constraints of high latency, low throughput and small frame sizes that characterize IEEE802.15.4 TSCH.

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 April 23, 2017.

Copyright Notice

Table of Contents

1. Introduction

Enrollment of new nodes into LLNs present unique challenges. The constrained nodes has no user interfaces, and even if they did, configuring thousands of such nodes manually is undesireable from a human resources issue, as well as the difficulty in getting consistent results.

This document is about a standard way to introduce new nodes into a 6tisch network that does not involve any direct manipulation of the nodes themselves. This act has been called "zero-touch" provisioning, and it does not occur by chance, but requires coordination between the manufacturer of the node, the service operator running the LLN, and the installers actually taking the devices out of the shipping boxes.

In other installations (such as some factory/industrial settings, and for some utilities), it is possible to pass devices through a "provisioning" room of some kind where the device in factory default state may be touched once (via a cable, or a push button, or by being placed in a faraday cage, etc.) such that the devices can be initialized in a way specific to that installation. The devices are then returned to inventory, and may be deployed at some future time. The node is not provisioned with the current keying material, as the network will need to be regularly rekeyed (even the algorithms may change!), so in the one-touch provisioning case, the goal is simply to introduce some elements into the new node (the "pledge") such that the enrollment process will take less energy and fewer network resources.

1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

The following terminology is repeated from [I-D.ietf-anima-bootstrapping-keyinfra] so as to have a common way to speak:

drop ship The physical distribution of equipment containing the "factory default" configuration to a final destination. In zero-touch scenarios there is no staging or pre-configuration during drop-ship.

imprint the process where a device obtains the cryptographic key material to identity and trust future interactions with a network. This term is taken from Konrad Lorenz's work in biology with new ducklings: during a critical period, the duckling would assume that anything that looks like a mother duck is in fact their mother. An equivalent for a device is to obtain the fingerprint of the network's root certification authority certificate. A device that imprints on an attacker suffers a similar fate to a duckling that imprints on a hungry wolf. Securely imprinting is a primary focus of this document. [duckling] anticipates this use.

enrollment the process where a device presents key material to a network and acquires a network specific identity. For example when a certificate signing request is presented to a certification authority and a certificate is obtained in response.

pledge the prospective device, which has the identity provided to at the factory. Neither the device nor the network knows if the device yet knows if this device belongs with this network. This is definition 6, according to [pledge].

Audit Token A signed token from the manufacturer authorized signing authority indicating that the bootstrapping event has been successfully logged. This has been referred to as an "authorization token" indicating that it authorizes bootstrapping to proceed.

Ownership Voucher A signed voucher from the vendor vouching that a specific domain "owns" the new entity as defined in [I-D.ietf-netconf-zerotouch].

1.2. Credentials

In the zero-touch scenario, every device expected to be drop shipped would have an [ieee802-1AR] manufacturer installed certificate (MIC). The private key part of the certificate would either be generated in the device, or installed securely (and privately) as part of the manufacturer process. [cullenCiscoPhoneDeploy] provides an example of process which has been active for a good part of a decade.

The MIC would be signed by the manufacturer's CA, the public key component of that would be included in the firmware.

1.2.1. One-Touch Assumptions

In a one-touch scenario, devices would be provided with some mechanism by which a secure association may be made in a controlled environment. There are many ways in which this might be done, and

detailing any of them is out of scope for this document. But, some notion of how this might be done is important so that the underlying assumptions can be reasoned about.

Some examples of how to do this could include: * JTAG interface * serial (craft) console interface * pushes of physical buttons simulatenous to network attachment * insecured devices operated in a Faraday cage

There are likely many other ways as well. What is assumed is that there can be a secure, private conversation between the Join Coordination Entity, and the Pledge, and that the two devices can exchange some trusted bytes of information.

1.2.2. Factory provided credentials (if any)

When a manufacturer installed certificate is provided as the IDevID, it SHOULD contain a number of fields. [I-D.ietf-anima-bootstrapping-keyinfra] provides a detailed set of requirements.

A manufacturer unique serial number MUST be provided in the serialNumber SubjectAltName extension, and MAY be repeated in the Common Name. There are no sequential or numeric requirements on the serialNumber, it may be any unique value that the manufacturer wants to use. The serialNumber SHOULD be printed on the packaging and/or on the device in a discrete way.

1.2.3. Credentials to be introduced

The goal of the bootstrap process is to introduce one or more new locally relevant credentials:

1. a certificate signed by a local certificate authority/registrar. This is the LDevID of [ieee802-1AR].

2. alternatively, a network-wide key to be used to secure L2 traffic.

3. alternatively, a network-wide key to be used to authenticate per-peer keying of L2 traffic using a mechanism such as provided by [ieee802159].

1.3. Network Assumptions

This document is about enrollment of constrained devices [RFC7228] to a constrained network. Constrained networks is such as [ieee802154], and in particular the time-slotted, channel hopping (tsch) mode,

feature low bandwidths, and limited opportunities to transmit. A key feature of these networks is that receivers are only listening at certain times.

1.3.1. Security above and below IP

802.15.4 networks have three kinds of layer-2 security:

- a network key that is shared with all nodes and is used for unicast and multicast.
- a series of network keys that are shared (agreed to) between pairs of nodes (the per-peer key)
- a network key that is shared with all nodes (through a group key management system), and is used for multicast traffic only

Setting up the credentials to bootstrap one of these kinds of security, (or directly configuring the key itself for the first case) is required. This is the security below the IP layer.

Security is required above the IP layer: there are three aspects which the credentials in the previous section are to be used.

- to provide for secure connection with a Path Computation Element [RFC4655], or other LLC (see ({RFC7554}} section 3).
- to initiate a connection between a Resource Server (RS) and an application layer Authorization Server (AS and CAS from [I-D.ietf-ace-actors]).

1.3.1.1. Perfect Forward Secrecy

Perfert Forward Secrecy (PFS) is the property of a protocol such that complete knowledge of the crypto state (for instance, via a memory dump) at time X does not imply that data from a disjoint time Y can also be recovered. ([PFS]).

PFS is important for two reasons: one is that it offers protection against the compromise of a node. It does this by changing the keys in a non-deterministic way. This second property also makes it much easier to remove a node from the network, as any node which has not participated in the key changing process will find itself no longer connected.

1.3.2. Join network assumptions

The network which the new pledge will connect to will have to have the following properties:

- a known PANID. The PANID 0xXXXX where XXXX is the assigned RFC# for this document is suggested.
- a minimal schedule with some Aloha time. This is usually in the same slotframe as the Extended Beacon, but a pledge MUST listen for an unencrypted Extended Beacon to so that it can synchronize.
- a known K1 key, as described in [I-D.ietf-6tisch-minimal] section 10, and used for reasons similar to [iec62591].

1.3.3. Number and cost of round trips

1.3.4. Size of packets, number of fragments

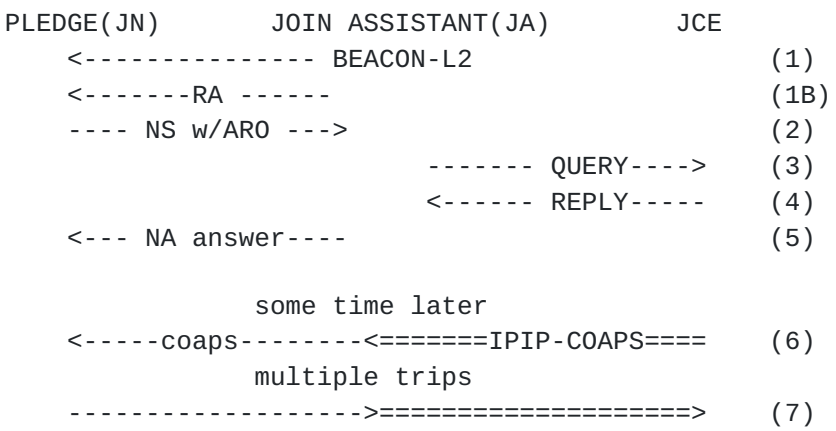
1.4. Target end-state for join process

2. Join Protocol

This section describes the interaction between a new pledge and the Join Assistant.

2.1. Diagram of Join Process

This time sequence diagram intentionally shows additional layer-2 and layer-3 interactions, in order to put the entire process into context.



2.2. Description of Pledge States in Join Process

2.2.1. (1) Layer-2 Enhanced Beacon

A new pledge must listen for a beacon for a period of at least 2s (HELP) on each of the 16 possible channels. The pledge SHOULD collect all beacons from heard on all channels before selecting a beacon to start with. If the pledge is unable to record all of the beacons that it hears due to limitations on volatile storage, then it should at least attempt to record the detail as to how to find that beacon again (channel, time sequence).

This search process entails having the pledge keep the receiver portion of it's radio active for the entire period of time.

The selection of which beacon to start with is outside the scope of this document. Implementers SHOULD make use of information such as: whether the L2 address of the Beacon has been tried before, whether a Router Advertisement IE is present, any Network Identifier [I-D.richardson-6lo-ra-in-ie] seen, and the strength of the signal.

Once a candidate network has been selected, the pledge can transition into a low-power duty cycle, waking only when the provided schedule indicates ALOHA slots which the pledge may use for the join process.

2.2.2. (1B) Layer-3 Router Advertisement

If [I-D.richardson-6lo-ra-in-ie] has not been used to embed a router advertisement in the Enhanced Beacon, then the pledge MUST wait to hear a Router Advertisement to know the layer-3 address of the adjacent router. These will be broadcast periodically during the ALOHA slot.

A pledge MAY timeout and send a layer-2 unicast Router Solicitation (to the layer-2 of the Enhanced Beacon) to the layer-3 all-routers address. A pledge MAY also take this timeout to mean that this router is unwilling to perform Join Assistant activities and the pledge should move on to another Enhanced Beacon.

2.2.3. (2) Pledge sends (unicast) Neighbor Solicitation to Join Assistant

This NS message is formed much like a Duplicate Address Detection (DAD) message described in [RFC6775] section-4.3: it is a solicitation by the pledge for it's own address. [RFC6775] does not describe doing DAD for link-local address however, so this aspect is new.

The Join Assistant will not validate the uniqueness using the DAR/DAC mechanism, but will otherwise process the NS as per normal: populating neighbor cache entries. The Join Assistant will take extra care with expiring neighbor cache entries: unsecured NS should never push secured NCE entries out of the cache or overwrite them. There are two equivalent ways to do this:

1. marking the origin of the NCE and limiting unsecured ones to some portion of the entries;

2. by considering unsecured NS to be arriving from a different virtual interface (different if_index) than secured ones. NCEs from different interfaces SHOULD already not be mixed.

The pledge SHOULD NOT have configured a short Layer-2 address as it has no way to allocate a non-duplicate short address. It SHOULD have formed a standard 64-bit layer-3 link-local address using a built-in IID. This IID MUST be placed into the Address Resolution option (ARO) option in the Neighbor Soliciation, as it serves as the index by which the domain registrar will use to identify the device.

The IID MAY be related to the layer-2 address, but privacy considerations recommend that the IID SHOULD instead be a form a stable privacy address [RFC7217]. Whichever method is used MUST be decided at manufacturing time, as the IID is also repeated as the SerialNumber in the Manufacturer Installed Certificate (MIC), also referred to as the 802.1AR IDevID.

2.2.4. (3) Join Assistant sends Query to Registar

This step does not involve the pledge, and it is described in section (#jastates).

2.2.5. (4) Join Assistant receives Acceptance response from Registrar

This step does not involve the pledge, and it is described in section (#jastates).

2.2.6. (5) Pledge receives (unicast) Neighbor Advertisement from join assistant

This NA message is again identical to the Duplicate Address Detection mechanism described in [RFC6775]. The status field of the ARO is extended (see (#ianaconsiderations)) to include an additional status value ND_NS_JOIN_DECLINED.

The pledge, upon receipt of ND_NS_JOIN_DECLINED considers that this network is not an appropriate network to join, and SHOULD move on to

attempt other networks. The pledge MUST also realize that this NA message MAY have been forced, and it SHOULD attempt joining this network again at a future time, but MUST NOT repeatedly attempt to join the same network.

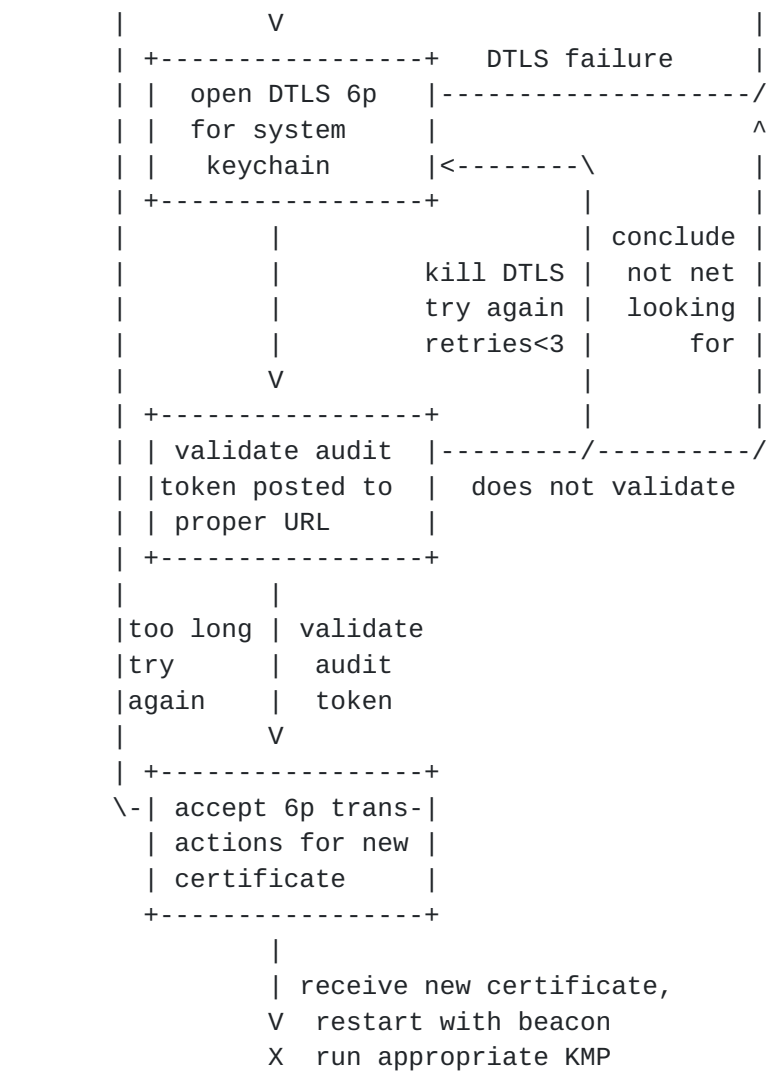
The pledge, upon receipt of a Neighbor Cache Full response SHOULD attempt to join using a different join assistant on the same network.

The pledge, upon receipt of a Duplicate Address response SHOULD attempt to join using a different join assistant on a different network, if it has such offers. If it has no such offers, it should wait at least NEIGHBOR-CACHE TIMEOUT, and then retry. This may be a sign of a Denial of Service attack, or it may be a non-malicious mis-configuration.

Upon receive of a successful NA, the pledge SHOULD consider that it is now in enrolled in a join queue. The pledge SHOULD resend Neighbor Soliciation (NUD) messages periodically as described in [RFC6775] to maintain the neighbor cache entry.

A pledge with other Join Assistant offers MAY abandon this Join Assistant after a period of XXX minutes and attempt to join using a different Join Assistant.

2.3. Join process state machine for pledge



2.4. Description of Join Assistant States in Join Process

The Join Assistant is a standard 6LR. It processes packets as described in [RFC6775], [I-D.ietf-6tisch-minimal] from secured (encrypted) sources.

In particular the maintenance of Neighbor Cache Entries as described in [RFC6775] section 3.5. The Join Assistant maintains two sets of NCE for each physical interface that it has: one set is for secured neighbors, and the other is for new pledges that wish to join. The storage allocated for pledges will generally be a small fraction of available space. The Join Assistant will garbage collect the different caches according to different thresholds. It MAY chose to free space from the insecure cache to make space for additional secure entries, but it MUST NOT do the opposite.

It sends Enhanced Beacons which are authenticated with the network-wide key ("K2"), but it does not encrypt the Beacons.

In addition, it listen for packets "encrypted" with the well-known "K1" key, and when it receives them, it considers them to be as "Insecure Packets". It MAY also accept unencrypted, unauthenticated packets as being "Insecure Packets"

Non-Join Assistant 6LRs would never accept K1 packets, nor unauthenticated packets. Normal 6LRs and hosts MUST accept unencrypted Enhanced Beacons which can be authenticated with the "K2" key.

In addition to seperating the secured and insecured packets for inbound processing, the Join Assistant also allocates a unique IID for the insecured interface. This IID is used to configure the Link Local address on that virtual interface. This Link Local address is called the Insecure Join Assistant Link Local, or IJALL.

In addition, this IID is combined with the global prefix(es) (as found in various PIO(s) from the routing protocols). This additional address is configured as an alias on the loopback interface such that the Join Assistant can receive packets to this address via secured network. This activity SHOULD generate a routing protocol update (such as an updated DAO). This IID SHOULD be generated using a stable privacy address mechanism as described in [RFC7217]. The easiest is to assign the insecure virtual interface a unique if_index. This new address is called the Pledge Tunnel End Point Address, or PTEPA.

2.4.1. Processing of Insecure Packets

Only the following insecure packets are to be accepted by the Join Assistant:

1. Unicast Neighbor Solicitations.

2. Unicast Link-Local UDP packets with a destination port that map to the Join Assistant's IPIP proxy.

2.4.1.1. Processing of Insecure Neighbor Solicitation packets

Upon receipt of an insecure unicast NS with an ARO option, the Join Assistant looks up an NCE by the IID contained in the ARO in the insecure cache. If it finds an existing there are three possibilities:

1. a lookup for this entry has previously been completed, and has resulted in a negative result. In this case, a negative ND_NS_JOIN_DECLINED NA is returned. The Join Assistant SHOULD rate limit the number of these messages that it is willing to return.

2. a lookup for this entry has previously been done, and has resulted in a positive result. The NCE entry should be "refresh", to keep it in the cache for a longer period of time, and a new NA returned with a positive status.

3. a lookup for this entry has previously been started, but no result has been received. In this case the Join Assistant SHOULD remain silent. The Join Assistant may wish to send a GRASP M_NOOP message to verify that the connection is still useable if it has not receive any traffic in some time.

If it does not find an existing entry, and there is space for another entry, (or it can make space via garbage collection), then a new entry is created, marked to be "in progress", and a new GRASP 6JOIN query is initiated, see section (#6joingrasp).

3. Join Assistant to Registrar protocol

There are three aspects to the protocols that the Join Assistant uses to communicate about its needs. They are:

1. Discovery of Registrar

2. Notification of new pledge to Registrar

3. Passing of traffic from Pledge to Registar

3.1. Discovery of Registrar

The address of the registrar MAY be determined by other protocols, such as DHCP, RA or RPL extensions, or provisioned into the Join Assistant via other configuration protocol such as 6p.

In order to support fully autonomic operations, the Join Assistant MAY use a GRASP discovery ([I-D.ietf-anima-grasp]) to find the address of the Registrar. [I-D.richardson-anima-6join-discovery] describes the details of the process.

In 6tisch networks multicast is not always available, requiring additional protocol [RFC7731] effort. Instead of doing a multicast GRASP discovery, the Join Assistant SHOULD instead to a TCP connect to the GRASP_LISTEN_PORT on the IP address of the DODAG root (when

RPL is used as the routing protocol for 6tisch), or the ABRO address when another protocol is used. The Join Assistant should then issue the appropriate M_DISCOVERY method using the 6JOIN objective. The GRASP discovery will then reply using the same TCP connection as per Unicast Discovery in [I-D.ietf-anima-grasp] section TBD.

3.2. Notification of a new pledge to Registar

As illustrated in (#joindiagram), when the Join Assistant receives a Neighbor Solication from a pledge, it must notify the Registar of the pledge, indicating to it how to reach the new pledge. The Registrar will respond with a positive acknowledgement if the Registrar is willing/able to accept the pledge. The Registrar will respond with a negative acknowledgement if the provided pledge identity (the IID in the ARO) is not one that the Registrar recognizes as belonging to this network.

The Registrar runs an ASA which is called the 6JOIN ASA (which can be discovered above). This query/response is done using GRASP with the discovered ASA process.

The query process is described in CDDL as:

```
request-6join-query = [M_REQ_NEG, session-id, "6JOIN", [IID, "6p-ipip"]]
IID = bytes .size 8
```

The response from the Registrar is describe as:

response-6join-query = [M_END, session-id, [O_ACCEPT]]

or for a negative response:

response-6join-query = [M_END, session-id, [O_DECLINE]]

for the 6p-ipip, the Registrar will need to know where to send the IPIP packets to. The Join Assistant will initiate the TCP connection to the Registrar's ASA using the IPv6 address associated with the insecure interface on which the pledge is located, i.e. using the PTEPA.

3.3. Passing of traffic from Pledge to Registar

When the Registrar is ready to initiate the pledge into the domain, the Registrar will reach out to the pledge using a secure CoAP protocol (6p). The security is provided using DTLS or EDHOC. As the pledge has only a link-local address, and the Registrar is not co-located on the same layer-2 as the pledge, the traffic must be relayed through the Join Assistant.

To do this the Registrar needs to configure a Link Local address on a virtual inteface which is the same as the PTEPA derived address.

The Registrar then sends it's traffic (UDP packets with CoAPS inside), inside of an IPIP header to the Join Assistant. The outer IP header is from the Registrar to the PTEPA. The inner IP is from the link local address configured above, and the destination is the Link Local address of the pledge.

The Registrar knows the IJALL by taking the IID from the connection address above, and knows the Link Local of the pledge from the IID in the objective message.

The Join Assistant, upon receipt of the IPIP traffic from the Registar on it's PTEPA, then decapsulates it and forwards it on the appropriate. (This is identical code to decapsulation of IPIP headers as specified in [I-D.ietf-roll-useofrplinfo].

The Join Assistant, upon receiving traffic from the pledge to the IJALL, it encapsulates it into an IPIP header, setting the source of this outer header to the PTEPA, and the destination being the Registrar.

The Join Assistant can do this for as many pledges as the Registrar decides to communicate with, without using any additional per-pledge state other than the obligatory Neighbor Cache Entries needed to translate L3 addresses to L2 addresses.

4. Privacy Considerations

[I-D.ietf-6lo-privacy-considerations] details a number of privacy considerations important in Resource Constrained nodes. There are two networks and three sets of constrained nodes to consider. They are: 1. the production nodes on the production network. 2. the new pledges, which have yet to enroll, and which are on a join network. 3. the production nodes which are also acting as proxy nodes.

4.1. Privacy Considerations for Production network

The details of this are out of scope for this document.

4.2. Privacy Considerations for New Pledges

New Pledges do not yet receive Router Advertisements with PIO options, and so configure link-local addresses only based upon layer-2 addresses using the normal SLAAC mechanisms described in [RFC4191].

These link-local addresses are visible to any on-link eavesdropper (who is synchronized to the same Join Assistant), so regardless of what is chosen they can be seen. This link-layer traffic is encapsulated by the Join Assistant into IPIP packets and carried to the JCE. The traffic SHOULD never leave the operator's network, and no outside traffic should enter, so it should not be possible to do any ICMP scanning as described in [I-D.ietf-6lo-privacy-considerations].

The join process described herein requires that some identifier meaningful to the network operator be communicated to the JCE via the Neighbor Advertisement's ARO option. This need not be a manufacturer created EUI-64 as assigned by IEEE; it could be another value with higher entropy and less interesting vendor/device information. Regardless of what is chosen, it can be used to track where the device attaches.

For most constrained device, network attachment occurs very infrequently, often only once in their lifetime, so tracking opportunities may be rare.

Further, during the enrollment process, a DTLS connection connection will be created. Unless TLS1.3 is used, the device identity will be visible to passive observers in the 802.11AR IDevID certificate that is sent. Even when TLS1.3 is used, an active attacker could collect the information by simply connecting to the device; it would not have to successful complete the negotiation either, or even attempt to Man-In-The-Middle the device.

There is, at the same time, significant value in avoiding a link-local DAD process by using an IEEE assigned EUI-64, and there is also significant advantage to the operator being able to see what the vendor of the new device is.

4.2.1. EUI-64 derived address for join time IID

It is therefore suggested that the IID used in the link-local address used during the join process be a vendor assigned EUI-64. After the join process has concluded, the device SHOULD be assigned a unique randomly generated long address, and a unique short address (not based upon the vendor EUI-64) for use at link-layer. At that point, all layer-3 content is encrypted by the layer-2 key.

4.3. Privacy Considerations for Join Assistant

5. Security Considerations

6. IANA Considerations

This document allocates one value from the subregistry "Address Registration Option Status Values": ND_NS_JOIN_DECLINED Join Assistant, JOIN DECLINED (TBD-AA)

7. Protocol Definition

8. References

8.1. Normative References

[cullenCiscoPhoneDeploy]
Jennings, C., "Transitive Trust Enrollment for Constrained Devices", 2012, <http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/papers/CullenJennings.pdf>.

[I-D.ietf-6lo-privacy-considerations]
Thaler, D., "Privacy Considerations for IPv6 over Networks of Resource-Constrained Nodes", draft-ietf-6lo-privacy-considerations-03 (work in progress), September 2016.

[I-D.ietf-6tisch-minimal]
Vilajosana, X. and K. Pister, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-16 (work in progress), June 2016.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., and S. Bjarnason, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-03 (work in progress), June 2016.

[I-D.ietf-anima-grasp]
Bormann, C., Carpenter, B., and B. Liu, "A Generic Autonomic Signaling Protocol (GRASP)", draft-ietf-anima-grasp-07 (work in progress), September 2016.

[I-D.ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch-09 (work in progress), July 2016.

[I-D.richardson-6lo-ra-in-ie]
Richardson, M., "802.15.4 Informational Element encapsulation of ICMPv6 Router Advertisements", draft-richardson-6lo-ra-in-ie-00 (work in progress), October 2016.

[I-D.richardson-anima-6join-discovery]
Richardson, M., "GRASP discovery of Registrar by Join Assistant", draft-richardson-anima-6join-discovery-00 (work in progress), October 2016.

[iec62591]
IEC, ., "62591:2016 Industrial networks - Wireless communication network and communication profiles - WirelessHART", 2016, <https://webstore.iec.ch/publication/24433>.

[ieee802-1AR]
IEEE Standard, ., "IEEE 802.1AR Secure Device Identifier", 2009, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[ieee802154]
IEEE Standard, ., "802.15.4-2015 - IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015, <http://standards.ieee.org/findstds/standard/802.15.4-2015.html>.

[ieee802159]
IEEE Standard, ., "802.15.9-2016 - IEEE Approved Draft Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams", 2016, <http://standards.ieee.org/findstds/standard/802.15.9-2016.html>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <http://www.rfc-editor.org/info/rfc7217>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

8.2. Informative References

[duckling] Stajano, F. and R. Anderson, "The resurrecting duckling: security issues for ad-hoc wireless networks", 1999, <https://www.cl.cam.ac.uk/~fms27/papers/1999-StajanoAnd-duckling.pdf>.

[I-D.ietf-ace-actors] Gerdes, S., Seitz, L., Selander, G., and C. Bormann, "An architecture for authorization in constrained environments", draft-ietf-ace-actors-04 (work in progress), September 2016.

[I-D.ietf-roll-useofrplinfo] Robles, I., Richardson, M., and P. Thubert, "When to use RFC 6553, 6554 and IPv6-in-IPv6", draft-ietf-roll-useofrplinfo-08 (work in progress), September 2016.

[PFS] Wikipedia, ., "Forward Secrecy", August 2016, <https://en.wikipedia.org/w/index.php?title=Forward_secrecy&oldid=731318899>.

[pledge] Dictionary.com, ., "Dictionary.com Unabridged", 2015, <http://dictionary.reference.com/browse/pledge>.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <http://www.rfc-editor.org/info/rfc4191>.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <http://www.rfc-editor.org/info/rfc4655>.

[RFC7554] Watteyne, T., Ed., Palattella, M., and L. Grieco, "Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement", RFC 7554, DOI 10.17487/RFC7554, May 2015, <http://www.rfc-editor.org/info/rfc7554>.

[RFC7731] Hui, J. and R. Kelsey, "Multicast Protocol for Low-Power and Lossy Networks (MPL)", RFC 7731, DOI 10.17487/RFC7731, February 2016, <http://www.rfc-editor.org/info/rfc7731>.

Appendix A. appendix

insert appendix here

Author's Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca