ANIMA WG
Internet-Draft
Intended status: Informational
Expires: September 18, 2016

M. Pritikin
Cisco
M. Richardson
SSW
M. Behringer
S. Bjarnason
Cisco

March 17, 2016

Bootstrapping Key Infrastructures
draft-ietf-anima-bootstrapping-keyinfra-02

Abstract

This document specifies automated bootstrapping of a key infrastructure (BSKI) using vendor installed IEEE 802.1AR manufacturing installed certificates, in combination with a vendor based service on the Internet. Before being authenticated, a new device has only link-local connectivity, and does not require a routable address. When a vendor provides an Internet based service, devices can be forced to join only specific domains but in limited/ disconnected networks or legacy environments we describe a variety of options that allow bootstrapping to proceed.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

To literally "pull yourself up by the bootstraps" is an impossible action. Similarly the secure establishment of a key infrastructure without external help is also an impossibility. Today it is accepted that the initial connections between nodes are insecure, until key distribution is complete, or that domain-specific keying material is pre-provisioned on each new device in a costly and non-scalable manner. This document describes a zero-touch approach to bootstrapping an entity by securing the initial distribution of key material using third-party generic keying material, such as a manufacturer installed IEEE 802.1AR certificate [IDevID], and a corresponding third-party service on the Internet.

The two sides of an association being bootstrapped authenticate each other and then determine appropriate authorization. This process is described as four distinct steps between the existing domain and the new entity being added:

o New entity authentication: "Who is this? What is its identity?"

o New entity authorization: "Is it mine? Do I want it? What are the chances it has been compromised?"

o Domain authentication: "What is this domain's claimed identity?"

o Domain authorization: "Should I join it?"

A precise answer to these questions can not be obtained without leveraging some established key infrastructure(s). A complexity that this protocol deals with are dealing with devices from a variety of

vendors, and a network infrastructure (the domain) that is operated by parties that do not have any priviledged relationship with the device vendors. The domain's decisions are based on the new entity's authenticated identity, as established by verification of previously installed credentials such as a manufacturer installed IEEE 802.1AR certificate, and verified back-end information such as a configured list of purchased devices or communication with a (unidirectionally) trusted third-party. The new entity's decisions are made according to verified communication with a trusted third-party or in a strictly auditable fashion.

Optimal security is achieved with IEEE 802.1AR certificates on each new entity, accompanied by a third-party Internet based service for verification. Bootstrapping concepts run to completion with less requirements, but are then less secure. A domain can choose to accept lower levels of security when a trusted third-party is not available so that bootstrapping proceeds even at the risk of reduced security. Only the domain can make these decisions based on administrative input and known behavior of the new entity.

The result of bootstrapping is that a domain specific key infrastructure is deployed. Since IEEE 802.1AR PKI certificates are used for identifying the new entity, and the public key of the domain identity is leveraged during communications with an Internet based service, which is itself authenticated using HTTPS, bootstrapping of a domain specific Public Key Infrastructure (PKI) is described. Sufficient agility to support bootstrapping alternative key infrastructures (such as symmetric key solutions) is considered although no such alternate key infrastructure is described.

1.1. 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].

The following terms are defined for clarity:

DomainID: The domain identity is the 160-bit SHA-1 hash of the BIT STRING of the subjectPublicKey of the domain trust anchor that is stored by the Domain CA. This is consistent with the RFC5280 Certification Authority subject key identifier of the Domain CA's self signed root certificate. (A string value bound to the Domain CA's self signed root certificate subject and issuer fields is often colloquially used as a humanized identity value but during protocol discussions the more exact term as defined here is used).

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.[imprinting].

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. Scope of solution

Questions have been posed as to whether this solution is suitable in general for Internet of Things (IoT) networks. In general the answer is no, but the terminology of [RFC7228] is best used to describe the boundaries.

The entire solution described in this document is aimed in general at non-constrained (i.e. class 2+) devices operating on a non-Challenged network. The entire solution described here is not intended to be useable as-is by constrained devices operating on challenged networks (such as 802.15.4 LLNs).

In many target applications, the systems involved are large router platforms with multi-gigabit inter-connections, mounted in controlled access data centers. But this solution is not exclusive to the large, it is intended to scale to thousands of devices located in hostile environments, such as ISP provided CPE devices which are drop-shipped to the end user. The situation where an order is fulfilled from distributed warehouse from a common stock and shipped directly to the target location at the request of the domain owner is explicitly supported. That stock ("SKU") could be provided to a number of potential domain owners, and the eventual domain owner will not know a-priori which device will go to which location.

Specifically, there are protocol aspects described here which might result in congestion collapse or energy-exhaustion of intermediate battery powered routers in an LLN. Those types of networks SHOULD NOT use this solution. These limitations are predominately related to the large credential and key sizes required for device authentication. Defining symmetric key techniques that meet the operational requirements is out-of-scope but the underlying protocol operations (TLS handshake and signing structures) have sufficient algorithm agility to support such techniques when defined.

The imprint protocol described here could, however, be used by non-energy constrained devices joining a non-constrained network (for instance, smart light bulbs are usually mains powered, and speak 802.11). It could also be used by non-constrained devices across a non-energy constrained, but challenged network (such as 802.15.4).

Some aspects are in scope for constrained devices on challenged networks: the certificate contents, and the process by which the four questions above are resolved is in scope. It is simply the actual on-the-wire imprint protocol which is likely inappropriate.

1.3. Trust bootstrap

The imprint protocol results in a secure relationship between the domain registrar and the new device. If the new device is sufficiently constrained that the ACE protocol should be leveraged for operation, (see [I-D.ietf-ace-actors]), and the domain registrar is also the Client Authorization Server or the Authorization Server, then it may be appropriate to use this secure channel to exchange ACE tokens.

2. Architectural Overview

The logical elements of the bootstrapping framework are described in this section. Figure 1 provides a simplified overview of the

components. Each component is logical and may be combined with other components as necessary.

```
                                             .
                                             .+------------------------+
      +--------------Drop Ship-------------->.| Vendor Service         |
      |                                      .+------------------------+
      |                                      .| M anufacturer|         |
      |                                      .| A uthorized  |Ownership|
      |                                      .| S igning     |Tracker  |
      |                                      .| A uthority   |         |
      |                                      .+--------------+---------+
      |                                      ..............  ^
      V                                                      |
   +-------+     ............................................|...
   |       |     .                                           |  .
   |       |     .  +------------+       +-----------+       |  .
   |       |     .  |            |       |           |       |  .
   |       |     .  |            |       |           <-------+  .
   |       |     .  |   Proxy    |       | Registrar |          .
   |       <-------->            <------->           |          .
   | New   |     .  |            |       |           |          .
   | Entity|     .  +------------+       +-----+-----+          .
   |       |     .                             |                .
   |       |     .           +-----------------+----------+     .
   |       |     .           | Domain Certification       |     .
   |       |     .           | Authority                  |     .
   +-------+     .           | Management and etc         |     .
                 .           +----------------------------+     .
                 .                                              .
                 ................................................
                               "Domain" components
```

Figure 1

Domain: The set of entities that trust a common key infrastructure trust anchor. This includes the Proxy, Registrar, Domain Certificate Authority, Management components and any existing entity that is already a member of the domain.

Domain CA: The domain Certification Authority (CA) provides certification functionalities to the domain. At a minimum it provides certification functionalities to the Registrar and stores the trust anchor that defines the domain. Optionally, it certifies all elements.

Registrar: A representative of the domain that is configured, perhaps autonomically, to decide whether a new device is allowed to join the domain. The administrator of the domain interfaces with a Registrar to control this process. Typically a Registrar is "inside" its domain.

New Entity: A new device or virtual machine or software component that is not yet part of the domain.

Proxy: A domain entity that helps the New Entity join the domain. A Proxy facilitates communication for devices that find themselves in an environment where they are not provided connectivity until after they are validated as members of the domain. The New Entity is unaware that they are communicating with a proxy rather than directly with the Registrar.

MASA Service: A Manufacturer Authorized Signing Authority (MASA) service on the global Internet. The MASA provides a trusted repository for audit log information concerning privacy protected bootstrapping events.

Ownership Tracker An Ownership Tracker service on the global internet. The Ownership Tracker uses business processes to accurately track ownership of all devices shipped against domains that have purchased them. Although optional this component allows vendors to provide additional value in cases where their sales and distribution channels allow for accurately tracking of such ownership.

We assume a multi-vendor network. In such an environment there could be a MASA or Ownership Tracker for each vendor that supports devices following this document's specification, or an integrator could provide a MASA service for all devices. It is unlikely that an integrator could provide Ownership Tracking services for multiple vendors.

This document describes a secure zero-touch approach to bootstrapping a key infrastructure; if certain devices in a network do not support this approach, they can still be bootstrapped manually. Although manual deployment is not scalable and is not a focus of this document the necessary mechanisms are called out in this document to ensure such edge conditions are covered by the architectural and protocol models.

3. Functional Overview

Entities behave in an autonomic fashion. They discover each other and autonomically bootstrap into a key infrastructure delineating the autonomic domain. See [I-D.irtf-nmrg-autonomic-network-definitions] for more information.

This section details the state machine and operational flow for each of the main three entities. The New Entity, the Domain (primarily the Registrar) and the MASA service.

A representative flow is shown in Figure 2:

```
     +--------+         +-------+      +------------+     +------------+
     | New    |         | Proxy |      | Domain     |     | Vendor     |
     | Entity |         |       |      | Registrar  |     | Service    |
     |        |         |       |      |            |     | (Internet  |
     +--------+         +-------+      +------------+     +------------+
      |                     |                   |                    |
      |<-RFC3927 IPv4 adr   |                   |                    |
    or|<-RFC4862 IPv6 adr   |                   |                    |
      |                     |                   |                    |
      |-------------------->|                   |                    |
      | optional: mDNS query|                   |                    |
      | RFC6763/RFC6762     |                   |                    |
      |                     |                   |                    |
      |<--------------------|                   |                    |
      | mDNS broadcast      |                   |                    |
      | response or periodic|                   |                    |
      |                     |                   |                    |
      |<------------------->|<----------------->|                    |
      |         (d)TLS via the Proxy            |                    |
      |<--Registrar TLS server authentication---|                    |
  [PROVISIONAL accept of server cert]           |                    |
      P---IEEE 802.1AR client authentication--->|                    |
      P                     |                   |                    |
      P---Request Audit Token (include nonce)-->|                    |
      P                     |                   |                    |
      P                     |       /--->       |                    |
      P                     |       |      [accept device?]          |
      P                     |       |      [contact Vendor]          |
      P                     |       |           |--New Entity ID---->|
      P                     |       |           |--Domain ID-------->|
      P                     |       |           |--optional:nonce--->|
      P                     |       |           |     [extract DomainID]
      P                     |       |           |                    |
      P                     |    optional:      |     [update audit log]
      P                     |       |can        |                    |
```

```
      P                      |       |occur       |     optional: is   |
      P                      |       |in          |     an ownership   |
      P                      |       |advance     |     voucher available?
      P                      |       |            |                    |
      P                      |       |            |<-device audit log--|
      P                      |       |            |<-audit token-------|
      P                      |       |            |                    |
      P                      |       |            |<-optional: --------|
      P                      |       \---->       |  ownership voucher |
      P                      |                    |                    |
      P                      |       [verify audit log or voucher]     |
      P                      |                    |                    |
      P<--Audit token and/or ownership voucher--|                    |
  [verify response         ]|                    |                    |
  [verify provisional cert ]|                    |                    |
      |                      |                    |                    |
      |--------------------------------------->|                    |
      | Continue with RFC7030 enrollment         |                    |
      | using now bidirectionally authenticated |                    |
      | TLS session.         |                    |                    |
      |                      |                    |                    |
      |                      |                    |                    |
      |                      |                    |                    |
```

   Figure 2

3.1. Behavior of a New Entity

   A New Entity that has not yet been bootstrapped attempts to find a local domain and join it. A New Entity MUST NOT automatically initiate bootstrapping if it has already been configured.

   States of a New Entity are as follows:

```
                +--------------+
                |   Start      |
                |              |
                +------+-------+
                       |
                +------v-------+
                |  Discover    |
   +------------>              |
   |            +------+-------+
   |                   |
   |            +------v-------+
   |            |  Identity    |
   ^------------+              |
   | rejected   +------+-------+
   |                   |
   |            +------v-------+
   |            | Request      |
   |            | Join         |
   |            +------+-------+
   |                   |
   |            +------v-------+
   |            |  Imprint     |   Optional
   ^------------+              <--+Manual input
   | Bad Vendor +------+-------+
   | response          |
   |            +------v-------+
   |            |  Enroll      |
   ^------------+              |
   | Enroll     +------+-------+
   | Failure           |
   |            +------v-------+
   |            |  Being       |
   ^------------+  Managed     |
    Factory     +--------------+
    reset
```

Figure 3

State descriptions for the New Entity are as follows:

1. Discover a communication channel to the "closest" Registrar.

2. Identify itself. This is done by presenting an IEEE 802.1AR credentials to the discovered Registrar (via the Proxy) in a (d)TLS handshake. (Although the Registrar is also authenticated these credentials are only provisionally accepted at this time).

3. Requests to Join the discovered Registrar. The acceptable imprint methods are indicated along with a nonce ensuring that any responses can be associated with this particular bootstrapping attempt.

4. Imprint on the Registrar. This requires verification of the vendor service "Audit Token" or the validation of the vendor service "Ownership Voucher". Either of these responses contains sufficient information for the New Entity to complete authentication of the Registrar. (The New Entity can now finish authentication of the Registrar (d)TLS server certificate)

5. Enroll by accepting the domain specific information from the Registrar, and by obtaining a domain certificate from the Registrar using a standard enrollment protocol, e.g. Enrollment over Secure Transport (EST) [RFC7030].

6. The New Entity is now a member of, and can be managed by, the domain and will only repeat the discovery aspects of bootstrapping if it is returned to factory default settings.

The following sections describe each of these steps in more detail.

3.1.1. Discovery

The result of discovery is logically communication with a Proxy instead of a Domain Registrar but in such a case the proxy facilitates communication with the actual Domain Registrar in a manner that is transparent to the New Entity. Therefore or clarity a Proxy is always assumed.

To discover the Domain Bootstrap Server the New Entity performs the following actions in this order:

a. MUST: Obtains a local address using either IPv4 or IPv6 methods as described in [RFC4862] IPv6 Stateless Address AutoConfiguration or [RFC3927] Dynamic Configuration of IPv4 Link-Local Addresses.

b. MAY: Performs DNS-based Service Discovery [RFC6763] over Multicast DNS [RFC6762] searching for the service "_bootstrapks._tcp.local."

c. SHOULD: Listen for an unsolicited broadcast response as described in [RFC6762]. This allows devices to avoid announcing their presence via mDNS broadcasts and instead silently join a network by watching for periodic unsolicited broadcast responses.

d. MAY: Performs DNS-based Service Discovery [RFC6763] over normal DNS operations. In this case the domain is known so the service searched for is "_bootstrapks._tcp.example.com".

e. MAY: If no local bootstrapks service is located using the DNS-based Service Discovery methods the New Entity contacts a well known vendor provided bootstrapping server by performing a DNS lookup using a well known URI such as "bootstrapks.vendor-example.com".

Once a Registrar is discovered (technically a communication channel through a Proxy) the New Entity communicates with the Registrar using the bootstrapping protocol defined in Section 5. The current DNS services returned during each query is maintained until bootstrapping is completed. If bootstrapping fails and the New Entity returns to the Discovery state it picks up where it left off and continues attempting bootstrapping. For example if the first Multicast DNS _bootstrapks._tcp.local response doesn't work then the second and third responses are tried. If these fail the New Entity moves on to normal DNS-based Service Discovery.

Once all discovered services are attempted the device SHOULD return to Multicast DNS and keep trying. The New Entity may prioritize selection order as appropriate for the anticipated environment.

[[EDNOTE: An appropriate backoff or rate limiting strategy should be defined here such that the device doesn't flood the local network with queries. If the device were to eventually give up -- or at least have too long between attempts -- a power cycle would restart the backoff mechanism.]]

3.1.2. Identity

The New Entity identifies itself during the communication protocol handshake. If the client identity is rejected the New Entity repeats the Discovery process using the next proxy or discovery method available.

The bootstrapping protocol server is not authenticated. Thus this connection is provisional and all data received is untrusted until sufficiently validated even though it is over a (D)TLS connection. This is aligned with the existing provisional mode of EST [RFC7030] during s4.1.1 "Bootstrap Distribution of CA Certificates".

All security associations established are between the new device and the Bootstrapping server regardless of proxy operations.

3.1.3. Request Join

The New Entity POSTs a request to join the domain to the Bootstrapping server. This request contains a New Entity generated nonce and informs the Bootstrapping server which imprint methods the New Entity will accept.

As indicated in EST [RFC7030] the bootstrapping server MAY redirect the client to an alternate server. This is most useful in the case where the New Entity has resorted to a well known vendor URI and is communicating with the vendor's Registrar directly. In this case the New Entity has authenticated the Registrar using the local Implicit Trust Anchor database and can therefore treat the redirect URI as a trusted URI which can also be validated using the Implicit Trust Anchor database. Since client authentication occurs during the TLS handshake the bootstrapping server has sufficient information to apply appropriate policy concerning which server to redirect to.

The nonce ensures the New Entity can verify that responses are specific to this bootstrapping attempt. This minimizes the use of global time and provides a substantial benefit for devices without a valid clock.

3.1.4. Imprint

The domain trust anchor is received by the New Entity during the bootstrapping protocol methods in the form of either an Audit Token containing the domainID or an explicit ownership voucher. The goal of the imprint state is to securely obtain a copy of this trust anchor without involving human interaction.

The enrollment protocol EST [RFC7030] details a set of non-autonomic bootstrapping methods such as:

o using the Implicit Trust Anchor database (not an autonomic solution because the URL must be securely distributed),

o engaging a human user to authorize the CA certificate using out-of-band data (not an autonomic solution because the human user is involved),

o using a configured Explicit TA database (not an autonomic solution because the distribution of an explicit TA database is not autonomic),

o and using a Certificate-Less TLS mutual authentication method (not an autonomic solution because the distribution of symmetric key material is not autonomic).

This document describes additional autonomic methods:

MASA audit token Audit tokens are obtained by the Registrar from the MASA service and presented to the New Entity for validation. These indicate to the New Entity that joining the domain has been logged by a trusted logging server.

Ownership Voucher Ownership Vouchers are obtained by the Registrar from the MASA service and explicitly indicate the fully qualified domain name of the domain the new entity currently belongs to. The Ownership Voucher is defined in [I-D.ietf-netconf-zerotouch].

Since client authentication occurs during the TLS handshake the bootstrapping server has sufficient information to apply appropriate policy concerning which method to use.

An arbitrary basic configuration information package that is signed by the domain can be delivered alongside the Audit Token or ownership validation. This information is signed by the domain private keys and is a one time delivery containing information such as which enrollment server to communicate with and which management system to communicate with. It is intended as a limited basic configuration for these purposes and is not intended to deliver entire final configuration to the device.

If the autonomic methods fail the New Entity returns to discovery state and attempts bootstrapping with the next available discovered Registrar.

3.1.5. Enrollment

As the final step of bootstrapping a Registrar helps to issue a domain specific credential to the New Entity. For simplicity in this document, a Registrar primarily facilitates issuing a credential by acting as an RFC5280 Registration Authority for the Domain Certification Authority.

Enrollment proceeds as described in Enrollment over Secure Transport (EST) [RFC7030]. The New Entity contacts the Registrar using EST as indicated:

- The New Entity is authenticated using the IEEE 802.1AR credentials.

- The EST section 4.1.3 CA Certificates Response is verified using either the Audit Token which provided the domain identity -or-

- The EST server is authenticated by using the Ownership Voucher indicated fully qualified domain name to build the EST URI such that EST section 4.1.1 bootstrapping using the New Entity implicit Trust Anchor database can be used.

3.1.6. Being Managed

Functionality to provide generic "configuration" information is supported. The parsing of this data and any subsequent use of the data, for example communications with a Network Management System is out of scope but is expected to occur after bootstrapping enrollment is complete. This ensures that all communications with management systems which can divulge local security information (e.g. network topology or raw key material) is secured using the local credentials issued during enrollment.

The New Entity uses bootstrapping to join only one domain. Management by multiple domains is out-of-scope of bootstrapping. After the device has successfully joined a domain and is being managed it is plausible that the domain can insert credentials for other domains depending on the device capabilities.

See Section 3.5.

3.2. Behavior of a Proxy

The role of the Proxy is to facilitate communications. The Proxy forwards packets between the New Entity and the Registrar that has been configured on the Proxy. The Proxy does not terminate the (d)TLS handshake.

In order to permit the proxy functionality to be implemented on the maximum variety of devices the chosen mechanism SHOULD use the minimum amount of state on the proxy device. While many devices in the ANIMA target space will be rather large routers, the proxy function is likely to be implemented in the control plane CPU such a device, with available capabilities for the proxy function similar to many class 2 IoT devices.

The document [I-D.richardson-anima-state-for-joinrouter] provides a more extensive analysis of the alternative proxy methods.

3.2.1. CoAP connection to Registrar

The proxy MUST implement an IPIP (protocol 41) encapsulation function for CoAP traffic to the configured UDP port on the registrar. The proxy does not terminate the CoAP DTLS connection. [[EDNOTE: The choice of CoAP as the mandatory to implement protocol rather than

HTTP maximizes code reuse on the smallest of devices. Unfortunately this means this document will have to include the EST over CoAP details as additional sections. The alternative is to make 'HTTPS proxy' method the mandatory to implement and provide a less friendly environment for the smallest of devices. This is a decision we'll have to see addressed by the broader team.]]

As a result of the Proxy Discovery process in section Section 3.1.1, the port number exposed by the proxy does not need to be well known, or require an IANA allocation. The address and port of the Registrar will be discovered by the GRASP protocol inside the ACP. For the IPIP encapsulation methods, the port announced by the Proxy MUST be the same as on the registrar.

The IPIP encapsulation allows the proxy to forward traffic which is otherwise not to be forwarded, as the traffic between New Node and Proxy use IPv6 Link Local addresses.

If the Proxy device has more than one interface on which it offers the proxy function, then it must select a unique IP address per interface in order so that the proxy can stateless return the reply packets to the correct link.

3.2.2. HTTPS proxy connection to Registrar

The proxy SHOULD also provide one of: an IPIP encapsulation of HTTP traffic on TCP port TBD to the registrar, an HTTP proxy which accepts URLs that reach the Registrar, or a TCP circuit proxy that connects the New Node to the Registrar.

In order to make the HTTP choice above transparent to the New Node, the New Node will always initiate an HTTP connection, and will always send an appropriate CONNECT message to initiate an HTTPS connection to the registrar. [[EDNOTE: The CONNECT syntax is that the New Entity specifies the Registrar server in the CONNECT line. See RFC7231 s4.3.6. We wish the Proxy to override any value with the locally known-to-the-proxy Registrar address.]]

When the Proxy provides a circuit proxy to the Registrar the Registrar MUST accept HTTP connections, and be willing to perform an HTTP proxy (CONNECT) operation to itself, and then initiate HTTPS.

When the Proxy provides a stateless IPIP encapsulation to the Registrar, then the Registrar will have to perform IPIP decapsulation, remembering the originating outer IPIP source address in order to qualify the inner link-local address. Being able to connect a TCP (HTTP) or UDP (CoAP) socket to a link-local address with an encapsulated IPIP header requires API extensions beyond

[RFC3542] for UDP use, and requires a form of connection latching (see section 4.1 of [RFC5386] and all of [RFC5660], except that a simple IPIP tunnel is used rather than an IPsec tunnel).

3.3. Behavior of the Registrar (Bootstrap Server)

Once a Registrar is established it listens for new entities and determines if they can join the domain. The registrar delivers any necessary authorization information to the new device and facilitates enrollment with the domain PKI.

Registrar behavior is as follows:

```
Contacted by New Entity
        +
        |
+-------v----------+
| Entity           | fail?
| Authentication   +---------+
+-------+----------+         |
        |                    |
+-------v----------+         |
| Entity           | fail?   |
| Authorization    +--------->
+-------+----------+         |
        |                    |
+-------v----------+         |
| Claiming the     | fail?   |
| Entity           +--------->
+-------+----------+         |
        |                    |
+-------v----------+         |
| Log Verification | fail?   |
|                  +--------->
+-------+----------+         |
        |                    |
+-------v----------+    +----v-------+
| Forward          |    |            |
| Audit            |    | Reject     |
| token + config   |    | Device     |
| to the Entity    |    |            |
+------------------+    +------------+
```

Figure 4

3.3.1. Entity Authentication

The applicable authentication methods detailed in EST [RFC7030] are:

- o the use of an IEEE 802.1AR IDevID credential,
- o or the use of a secret that is transmitted out of band between the New Entity and the Registrar (this use case is not autonomic).

3.3.2. Entity Authorization

In a fully automated network all devices must be securely identified and authorized to join the domain.

A Registrar accepts or declines a request to join the domain, based on the authenticated identity presented. Automated acceptance criteria include:

- o allow any device of a specific type (as determined by the IEEE 802.1AR device identity),
- o allow any device from a specific vendor (as determined by the IEEE 802.1AR identity),
- o allow a specific device from a vendor (as determined by the IEEE 802.1AR identity)

Since all New Entities accept Audit Tokens the Registrar MUST use the vendor provided MASA service to verify that the device's history log does not include unexpected Registrars. If a device had previously registered with another domain, the Registrar of that domain would show in the log.

In order to validate the IEEE 802.1AR device identity the Registrar maintains a database of vendor trust anchors (e.g. vendor root certificates or keyIdentifiers for vendor root public keys). For user interface purposes this database can be mapped to colloquial vendor names. Registrars can be shipped with the trust anchors of a significant number of third-party vendors within the target market.

If a device is accepted into the domain, it is expected request a domain certificate through a certificate enrollment process. The result is a common trust anchor and device certificates for all autonomic devices in a domain (these certificates can subsequently be used to determine the boundaries of the homenet, to authenticate other domain nodes, and to autonomically enable services on the homenet). The authorization performed during this phase MAY be

cached for the TLS session and applied to subsequent EST enrollment requests so long as the session lasts.

3.3.3. Claiming the New Entity

Claiming an entity establishes an audit log at the MASA server and provides the Registrar with proof, in the form of a MASA authorization token, that the log entry has been inserted. As indicated in Section 3.1.4 a New Entity will only proceed with bootstrapping if a validated MASA authorization token has been received. The New Entity therefore enforces that bootstrapping only occurs if the claim has been logged. There is no requirement for the vendor to definitively know that the device is owned by the Registrar.

Registrar's obtain the Vendor URI via static configuration or by extracting it from the IEEE 802.1AR credential. The imprint method supported by the New Entity is known from the IEEE 802.1AR credential. [[EDNOTE: An appropriate extension for indicating the Vendor URI and imprint method could be defined using the methods described in [I-D.lear-mud-framework]]].

During initial bootstrapping the New Entity provides a nonce specific to the particular bootstrapping attempt. The Registrar SHOULD include this nonce when claiming the New Entity from the MASA service. Claims from an unauthenticated Registrar are only serviced by the MASA resource if a nonce is provided.

The Registrar can claim a New Entity that is not online by forming the request using the entities unique identifier and not including a nonce in the claim request. Audit Tokens obtained in this way do not have a lifetime and they provide a permanent method for the domain to claim the device. Evidence of such a claim is provided in the audit log entries available to any future Registrar. Such claims reduce the ability for future domains to secure bootstrapping and therefore the Registrar MUST be authenticated by the MASA service.

An ownership voucher requires the vendor to definitively know that a device is owned by a specific domain. The method used to "claim" this are out-of-scope. The Registrar simply requests an ownership validation token and the New Entity trusts the response.

3.3.4. Log Verification

The Registrar requests the log information for the new entity from the MASA service. The log is verified to confirm that the following is true to the satisfaction of the Registrar's configured policy:

- Any nonceless entries in the log are associated with domainIDs recognized by the registrar.

- Any nonce'd entries are older than when the domain is known to have physical possession of the new entity or that the domainIDs are recognized by the registrar.

If any of these criteria are unacceptable to the registrar the entity is rejected. The Registrar MAY be configured to ignore the history of the device but it is RECOMMENDED that this only be configured if hardware assisted NEA [RFC5209] is supported.

3.3.5. Forwarding Audit Token plus Configuration

The Registrar forwards the received Audit Token to the New Entity. To simplify the message flows an initial configuration package can be delivered at this time which is signed by a representative of the domain.

[[EDNOTE: format TBD. The configuration package signature data must contain the full certificate path sufficient for the new entity to use the domainID information (as a trust anchor) to accept and validate the configuration)]]

3.4. Behavior of the MASA Service

The MASA service is provided by the Factory provider on the global Internet. The URI of this service is well known. The URI SHOULD also be provided as an IEEE 802.1AR IDevID X.509 extension (a "MASA Audit Token Distribution Point" extension).

The MASA service provides the following functionalities to Registrars:

3.4.1. Issue Authorization Token and Log the event

A Registrar POSTs a claim message optionally containing the bootstrap nonce to the MASA server.

If a nonce is provided the MASA service responds to all requests. The MASA service verifies the Registrar is representative of the domain and generates a privacy protected log entry before responding with the Audit Token.

If a nonce is not provided then the MASA service MUST authenticate the Registrar as a valid customer. This prevents denial of service attacks.

3.4.2. Retrieve Audit Entries from Log

When determining if a New Entity should be accepted into a domain the Registrar retrieves a copy of the audit log from the MASA service. This contains a list of privacy protected domain identities that have previously claimed the device. Included in the list is an indication of the time the entry was made and if the nonce was included.

3.5. Leveraging the new key infrastructure / next steps

As the devices have a common trust anchor, device identity can be securely established, making it possible to automatically deploy services across the domain in a secure manner.

Examples of services:

- Device management.

- Routing authentication.

- Service discovery.

3.5.1. Network boundaries

When a device has joined the domain, it can validate the domain membership of other devices. This makes it possible to create trust boundaries where domain members have higher level of trusted than external devices. Using the autonomic User Interface, specific devices can be grouped into to sub domains and specific trust levels can be implemented between those.

3.6. Interactions with Network Access Control

The assumption is that Network Access Control (NAC) completes using the New Entity 802.1AR credentials and results in the device having sufficient connectivity to discovery and communicate with the proxy. Any additional connectivity or quarantine behavior by the NAC infrastructure is out-of-scope. After the devices has completed bootstrapping the mechanism to trigger NAC to re-authenticate the device and provide updated network privileges is also out-of-scope.

This achieves the goal of a bootstrap architecture that can integrate with NAC but does not require NAC within the network where it wasn't previously required. Future optimizations can be achieved by integrating the bootstrapping protocol directly into an initial EAP exchange.

4. Domain Operator Activities

This section describes how an operator interacts with a domain that supports the bootstrapping as described in this document.

4.1. Instantiating the Domain Certification Authority

This is a one time step by the domain administrator. This is an "off the shelf" CA with the exception that it is designed to work as an integrated part of the security solution. This precludes the use of 3rd party certification authority services that do not provide support for delegation of certificate issuance decisions to a domain managed Registration Authority.

4.2. Instantiating the Registrar

This is a one time step by the domain administrator. One or more devices in the domain are configured take on a Registrar function.

A device can be configured to act as a Registrar or a device can auto-select itself to take on this function, using a detection mechanism to resolve potential conflicts and setup communication with the Domain Certification Authority. Automated Registrar selection is outside scope for this document.

4.3. Accepting New Entities

For each New Entity the Registrar is informed of the unique identifier (e.g. serial number) along with the manufacturer's identifying information (e.g. manufacturer root certificate). This can happen in different ways:

1. Default acceptance: In the simplest case, the new device asserts its unique identity to the registrar. The registrar accepts all devices without authorization checks. This mode does not provide security against intruders and is not recommended.

2. Per device acceptance: The new device asserts its unique identity to the registrar. A non-technical human validates the identity, for example by comparing the identity displayed by the registrar (for example using a smartphone app) with the identity shown on the packaging of the device. Acceptance may be triggered by a click on a smartphone app "accept this device", or by other forms of pairing. See also [I-D.behringer-homenet-trust-bootstrap] for how the approach could work in a homenet.

3. Whitelist acceptance: In larger networks, neither of the previous approaches is acceptable. Default acceptance is not secure, and

a manual per device methods do not scale. Here, the registrar is provided a priori with a list of identifiers of devices that belong to the network. This list can be extracted from an inventory database, or sales records. If a device is detected that is not on the list of known devices, it can still be manually accepted using the per device acceptance methods.

4. Automated Whitelist: an automated process that builds the necessary whitelists and inserts them into the larger network domain infrastructure is plausible. Once set up, no human intervention is required in this process. Defining the exact mechanisms for this is out of scope although the registrar authorization checks is identified as the logical integration point of any future work in this area.

None of these approaches require the network to have permanent Internet connectivity. Even when the Internet based MASA service is used, it is possible to pre-fetch the required information from the MASA a priori, for example at time of purchase such that devices can enroll later. This supports use cases where the domain network may be entirely isolated during device deployment.

Additional policy can be stored for future authorization decisions. For example an expected deployment time window or that a certain Proxy must be used.

4.4. Automatic Enrollment of Devices

The approach outlined in this document provides a secure zero-touch method to enroll new devices without any pre-staged configuration. New devices communicate with already enrolled devices of the domain, which proxy between the new device and a Registrar. As a result of this completely automatic operation, all devices obtain a domain based certificate.

4.5. Secure Network Operations

The certificate installed in the previous step can be used for all subsequent operations. For example, to determine the boundaries of the domain: If a neighbor has a certificate from the same trust anchor it can be assumed "inside" the same organization; if not, as outside. See also Section 3.5.1. The certificate can also be used to securely establish a connection between devices and central control functions. Also autonomic transactions can use the domain certificates to authenticate and/or encrypt direct interactions between devices. The usage of the domain certificates is outside scope for this document.

5. Protocol Details

For simplicity the bootstrapping protocol is described as extensions to EST [RFC7030].

EST provides a bootstrapping mechanism for new entities that are configured with the URI of the EST server such that the Implicit TA database can be used to authenticate the EST server. Alternatively EST clients can "engage a human user to authorize the CA certificate using out-of-band data such as a CA certificate". EST does not provide a completely automated method of bootstrapping the PKI as both of these methods require some user input (either of the URI or authorizing the CA certificate).

This section details additional EST functionality that support automated bootstrapping of the public key infrastructure. These additions provide for fully automated bootstrapping. These additions are to be optionally supported by the EST server within the same .well-known URI tree as the existing EST URIs.

The "New Entity" is the EST client and the "Registrar" is the EST server.

The extensions for the client are as follows:

- o The New Entity provisionally accept the EST server certificate during the TLS handshake as detailed in EST section 4.1.1 ("Bootstrap Distribution of CA Certificates").

- o The Registrar requests and validates the Audit Token from the vendor authorized MASA service.

- o The New Entity requests and validates the Audit Token as described below. At this point the New Entity has sufficient information to validate domain credentials.

- o The New Entity calls the EST defined /cacerts method to obtain the current CA certificate. These are validated using the Audit Token.

- o The New Entity completes bootstrapping as detailed in EST section 4.1.1.

These extensions could be implemented as an independent protocol from EST but since the overlap with basic enrollment is extensive, particularly with respect to client authorization, they are presented here as additions to EST.

In order to obtain a validated Audit Token and Audit Log the
Registrar contacts the MASA service Service using REST calls:

```
           +-----------+ +----------+ +-----------+ +----------+
           | New       | |          | |           | |          |
           | Entity    | | Proxy    | | Registrar | | Vendor   |
           |           | |          | |           | |          |
           ++----------+ +--+-------+ +-----+-----+ +--------+-+
            |               |               |                |
            |               |               |                |
            | (D)TLS hello  |               |                |
Establish   +---------------> (D)TLS hello  |                |
(D)TLS      |               |--------------->                |
connection  |          (forwarding)         |                |
            | Server Cert   <---------------+                |
            <---------------+               |                |
            | Client Cert   |               |                |
            +------------------------------->                |
            |               |               |                |
HTTP REST   | POST /requestaudittoken       |                |
Data        +--------------------nonce------>                |
            |               .               | /requestaudittoken
            |               .               +---------------->
            |                               <----------------+
            |                               | /requestauditlog
            |                               +---------------->
            | audit token or owner voucher  <----------------+
            <-------------------------------+                |
            | (optional config information) |                |
            |               .               |                |
            |               .               |                |
```

Figure 5

In some use cases the Registrar may need to contact the Vendor in
advanced, for example when the target network is air-gapped. The
nonceless request format is provided for this and the resulting flow
is slightly different. The security differences associated with not
knowing the nonce are discussed below:

```
              +-----------+ +----------+ +-----------+ +----------+
              | New       | |          | |           | |          |
              | Entity    | | Proxy    | | Registrar | | Vendor   |
              |           | |          | |           | |          |
              ++----------+ +--+-------+ +-----+-----+ +--------+-+
               |               |               |                |
               |               |               |                |
               |               |               | /requestaudittoken
               |               |  (nonce       +---------------->
               |               |  unknown)     <----------------+
               |               |               | /requestauditlog
               |               |               +---------------->
               |               |               <----------------+
               | (D)TLS hello  |               |                |
   Establish   +---------------> (D)TLS hello  |                |
   (D)TLS      |               |--------------->                |
   connection  |          (forwarding)          |                |
               | SerVer Cert   <---------------+                |
               <---------------+               |                |
               | Client Cert   |               |                |
               +------------------------------->                |
               |               |               |                |
   HTTP REST   | POST /requestaudittoken       |                |
   Data        +----------------------nonce---->   (discard     |
               | audit token or owner Voucher  |   nonce)       |
               <-------------------------------+                |
               | (optional config information) |                |
               |               .               |                |
               |               .               |                |
```

Figure 6

5.1. IEEE 802.1AR as client identity

The Registrar authenticates the client and performs authorization checks to ensure this client is expected to join the domain. This require a common procedure for representing and verifying the identity of the client. The methods detailed in [RFC6125] such as matching DNS Domain Name or Application Service Type are not directly applicable.

Clients presents an IEEE 802.AR certificate complete with subject field identifying the device uniquely in the Distinguished Name serialNumber subfield. The subjectAltName MAY contain a hardwareModuleName as specified in RFC4108. The Registrar extracts this information and compares against a per vendor access control list. (This can be implemented with a single database table so long

as the authority key identifier is also maintained and checked to ensure that no two vendors collide in their use of serialNumber's).

When enrollment is complete and a local certificate is issued to the new device the local CA has complete control over the namespace. If this credential is intended for RFC6125 style TLS connections where servers are identified by a server's DNS-ID identity the CA is likely to ensure the dNSName field is populated. For Anima purposes the IEEE 802.1AR serialNumber and hardwareModuleName fields MUST be propagated to the issued certificate.

[[EDNOTE: the above authority key identifier trick works for database lookups and here the inclusion of the DNS name would serve the same purpose. Alternatively an Anima specified domain specific identifier must be indicated.]]

5.2. EST over CoAP

[[EDNOTE: In order to support smaller devices the above section on Proxy behavior introduces mandatory to implement support for CoAP support by the Proxy. This implies similar support by the New Entity and Registrar and means that the EST protocol operation encapsulation into CoAP needs to be described. EST is HTTP based and "CoaP is designed to easily interface with HTTP for integration" [RFC7252] so this section is anticipated to be relatively straightforward. A complexity is that the large message sizes necessary for bootstrapping will require support for [draft-ietf-core-block].]]

5.3. Request Audit Token

When the New Entity reaches the EST section 4.1.1 "Bootstrap Distribution of CA Certificates" state but wishes to proceed in a fully automated fashion it makes a request for a MASA authorization token from the Registrar.

This is done with an HTTPS POST using the operation path value of "/requestaudittoken".

The request format is JSON object containing a nonce.

Request media type: application/auditnonce

Request format: a JSON file with the following:

```
{"nonce":"<64bit nonce value>", "OwnershipValidation":boolean}
```

[[EDNOTE: exact format TBD. There is an advantage to having the client sign the nonce (similar to a PKI Certification Signing

Request) since this allows the MASA service to confirm the actual device identity. It is not clear that there is a security benefit from this since its the New Entity that verifies the nonce.]]

The Registrar validates the client identity as described in EST [RFC7030] section 3.3.2. The registrar performs authorization as detailed in Section 3.3.2. If authorization is successful the Registrar obtains an Audit Token from the MASA service (see Section 5.2).

The received MASA authorization token is returned to the New Entity.

As indicated in EST [RFC7030] the bootstrapping server can redirect the client to an alternate server. If the New Entity authenticated the Registrar using the well known URI method then the New Entity MUST follow the redirect automatically and authenticate the new Registrar against the redirect URI provided. If the New Entity had not yet authenticated the Registrar because it was discovered and was not a known-to-be-valid URI then the new Registrar must be authenticated using one of the two autonomic methods described in this document.

5.4. Request Audit Token from MASA

The Registrar requests the Audit Token from the MASA service using a REST interface. For simplicity this is defined as an optional EST message between the Registrar and an EST server running on the MASA service although the Registrar is not required to make use of any other EST functionality when communicating with the MASA service. (The MASA service MUST properly reject any EST functionality requests it does not wish to service; a requirement that holds for any REST interface).

This is done with an HTTP POST using the operation path value of "/requestaudittoken".

The request format is a JSON object optionally containing the nonce value (as obtained from the bootstrap request) and the IEEE 802.1AR identity of the device as a serial number (the full certificate is not needed and no proof-of-possession information for the device identity is included). The New Entity's serial number is extracted from the IEEE 802.1AR subject name:

```
{"nonce":"<64bit nonce value>", "serialnumber", "<subjectname/
subjectaltname serial number>"}
```

The Registrar MAY exclude the nonce from the request. Doing so allows the Registrar to request an authorization token when the New

Entity is not online, or when the target bootstrapping environment is not on the same network as the MASA server. If a nonce is not provided the MASA server MUST authenticate the client as described in EST [RFC7030] section 3.3.2. The registrar performs authorization as detailed in Section 3.3.2. If authorization is successful the Registrar obtains an Audit Token from the MASA service (see Section 5.4).

The JSON message information is encapsulated in a PKCS7 signed data structure that is signed by the Registrar. The entire certificate chain, up to and including the Domain CA, MUST be included in the PKCS7.

The MASA service checks the internal consistency of the PKCS7 but MAY not authenticate the domain identity information. The domain is not know to the MASA server in advance and a shared trust anchor is not implied. The MASA server MUST verify that the PKCS7 is signed by a Registrar certificate (by checking for the cmc-idRA field) that was issued by a the root certificate included in the PKCS7. This ensures that the Registrar is in fact an authorized Registrar of the unknown domain.

The domain ID (e.g. hash of the public key of the domain) is extracted from the root certificate and is used to populate the MASA authorization token and to update the audit log. The authorization token consists of the nonce, if supplied, the serialnumber and the domain identity:

```
{"nonce":"<64bit nonce value>", "serialnumber", "<subjectname/
subjectaltname serial number>","domainID":}
```

[[EDNOTE: There is a strong similarity between this and the previous section. Both involve requesting the Audit Token from the upstream element. Because there are differing requirements on the data submitted and the signing of that data they are specified in distinct sections. The design team should have a meeting to discuss how to unify these sections or make the distinctions more clear]]

5.5. Basic Configuration Information Package

When the MASA authorization token is returned to the New Entity an arbitrary information package can be signed and delivered along side it. This is signed by the Domain Registrar. The New Entity first verifies the Audit Token and, if it is valid, then uses the domain's TA to validate the Information Package.

[[EDNOTE: The domainID as included in the log and as sent in the authorization token is only a hash of the domain root certificate.

This is insufficient for the new entity to move out of the provisional state as it needs a full root certificate to validate the TLS certificate chain. This information package could be used to deliver the full certificate or the full certificate could be included in the authorization token. Lacking either the new entity needs to stay in the provisional state until it performs an RFC7030 /getcacerts to obtain the full certificate chain.]]

[[EDNOTE: The package format to be specified here. Any signed format is viable and ideally one can simply be specified from netconf. The Registar knows the New Entity device type from the 802.1AR credential and so is able to determine the proper format for the configuration.]]

5.6. Request MASA authorization log

A registrar requests the MASA authorization log from the MASA service using this EST extension.

This is done with an HTTP GET using the operation path value of "/requestMASAlog".

The log data returned is a file consisting of all previous log entries. For example:

```
"log":[
  {"date":"<date/time of the entry>"},
   "domainID":"<domainID as extracted from the root
                certificate within the PKCS7 of the
                audit token request>",
   "nonce":"<any nonce if supplied (or NULL)>"},

  {"date":"<date/time of the entry>"},
   "domainID":"<domainID as extracted from the root
                certificate within the PKCS7 of the
                audit token request>",
   "nonce":"<any nonce if supplied (or NULL)>"},
]
```

Distribution of a large log is less than ideal. This structure can be optimized as follows: All nonce-less entries for the same domainID can be condensed into the single most recent nonceless entry.

The Registrar uses this log information to make an informed decision regarding the continued bootstrapping of the New Entity. For example if the log includes unexpected domainIDs this is indicative of problematic imprints by the new entity. If unexpected nonce-less entries exist this is indicative of the permanent ability for the

unknown domain to trigger a reset of the device and take over management of it. Equipment that is purchased pre-owned can be expected to have an extensive history.

Log entries containing the Domain's ID can be compared against local history logs in search of discrepancies.

[[EDNOTE: certificate transparency style use of merkle tree hash's might offer an alternative log entry method]]

6. Reduced security operational modes

A common requirement of bootstrapping is to support less secure operational modes for support specific use cases. The following sections detail specific ways that the New Entity, Registrar and MASA can be configured to run in a less secure mode for the indicated reasons.

6.1. Trust Model

```
+--------+         +-------+      +------------+     +------------+
| New    |         | Proxy |      | Domain     |     | Vendor     |
| Entity |         |       |      | Registrar  |     | Service    |
|        |         |       |      |            |     | (Internet  |
+--------+         +-------+      +------------+     +------------+
```

Figure 7

New Entity: The New Entity could be compromised and providing an attack vector for malware. The entity is trusted to only imprint using secure methods described in this document. Additional endpoint assessment techniques are RECOMMENDED but are out-of-scope of this document.

Proxy: Provides proxy functionalities but is not involved in security considerations.

Registrar: When interacting with a MASA server the Registrar makes all decisions. When ownership vouchers are involved the Registrar is only a conduit and all security decisions are made on the vendor service.

Vendor Service, MASA: This form of vendor service is trusted to accurately log all claim attempts and to provide authoritative log information to Registrars. The MASA does not know which devices are associated with which domains. [[EDNOTE: these claims could be strengthened using by using cryptographic log techniques to

provide append only", cryptographic assured, publicly auditable logs. Current text provides for a fully trusted vendor.]]

Vendor Service, Ownership Validation: This form of vendor service is trusted to accurately know which device is owned by which domain.

6.2. New Entity security reductions

Although New Entity can choose to run in less secure modes this is MUST NOT be the default state because it permanently degrades the security for all other uses cases.

The device may have an operational mode where it skips Audit Token or Ownership Voucher validation one time. For example if a physical button is depressed during the bootstrapping operation. This can be useful if the vendor service is unavailable. This behavior SHOULD be available via local configuration or physical presence methods to ensure new entities can always be deployed even when autonomic methods fail. This allows for unsecure imprint.

It is RECOMMENDED that this only be available if hardware assisted NEA [RFC5209] is supported.

6.3. Registrar security reductions

The Registrar can choose to accept devices using less secure methods. These methods are RECOMMENDED when low security models are needed as the security decisions are being made by the local administrator:

1. The registrar MAY choose to accept all devices, or all devices of a particular type, at the administrator's discretion. This could occur when informing the Registrar of unique identifiers of new entities might be operationally difficult.

2. The registrar MAY choose to accept devices that claim a unique identity without the benefit of authenticating that claimed identity. This could occur when the New Entity does not include an IEEE 802.1AR factory installed credential.

3. The registrar MAY request nonce-less Audit Tokens from the MASA service. These tokens can then be transmitted to the Registrar and stored until they are needed during bootstrapping operations. This is for use cases where target network is protected by an air gap and therefore can not contact the MASA service during New Entity deployment.

4. The registrar MAY ignore unrecognized nonce-less Audit Log entries. This could occur when used equipment is purchased with

a valid history being deployed in air gap networks that required permanent Audit Tokens.

These modes are not available for devices that require a vendor Ownership Voucher. The methods vendors use to determine which devices are owned by which domains is out-of-scope.

6.4. MASA security reductions

Lower security modes chosen by the MASA service effect all device deployments unless bound to the specific device identities. In which case these modes can be provided as additional features for specific customers. The MASA service can choose to run in less secure modes by:

1. Not enforcing that a Nonce is in the Audit Token. This results in distribution of Audit Tokens that never expire and in effect makes the Domain an always trusted entity to the New Entity during any subsequent bootstrapping attempts. That this occurred is captured in the log information so that the Domain registrar can make appropriate security decisions when a New Entity joins the Domain. This is useful to support use cases where Registrars might not be online during actual device deployment. Because this results in long lived Audit Tokens and do not require the proof that the device is online this is only accepted when the Registrar is authenticated by the MASA server and authorized to provide this functionality. The MASA server is RECOMMENDED to use this functionality only in concert with Ownership Validation tracking.

2. Not verifying ownership before responding with an Audit Token. This is expected to be a common operational model because doing so relieves the vendor providing MASA services from having to tracking ownership during shipping and supply chain and allows for a very low overhead MASA service. The Registrar uses the audit log information as a defense in depth strategy to ensure that this does not occur unexpectedly (for example when purchasing new equipment the Registrar would throw an error if any audit log information is reported).

7. Security Considerations

In order to support a wide variety of use cases, devices can be claimed by a registrar without proving possession of the device in question. This would result in a nonceless, and thus always valid, claim. Or would result in an invalid nonce being associated with a claim. The MASA service is required to authenticate such Registrars but no programmatic method is provided to ensure good behavior by the

MASA service. Nonceless entries into the audit log therefore permanently reduce the value of a device because future Registrars, during future bootstrap attempts, would now have to be configured with policy to ignore previously (and potentially unknown) domains.

Future registrars are recommended to take the audit history of a device into account when deciding to join such devices into their network. If the MASA server were to have allowed a significantly large number of claims this might become onerous to the MASA server which must maintain all the extra log entries. Ensuring the Registrar is representative of a valid customer domain even without validating ownership helps to mitigate this.

It is possible for an attacker to send an authorization request to the MASA service directly after the real Registrar obtains an authorization log. If the attacker could also force the bootstrapping protocol to reset there is a theoretical opportunity for the attacker to use the Audit Token to take control of the New Entity but then proceed to enroll with the target domain. Possible prevention mechanisms include:

o Per device rate limits on the MASA service ensure such timing attacks are difficult.

o In the advent of an unexpectedly lost bootstrapping connection the Registrar repeats the request for audit log information.

As indicated in EST [RFC7030] the connection is provisional and untrusted until the server is successfully authorized. If the server provides a redirect response the client MUST follow the redirect but the connection remains provisional. If the client uses a well known URI for contacting a well known Registrar the EST Implicit Trust Anchor database is used as is described in RFC6125 to authenticate the well known URI. In this case the connection is not provisional and RFC6125 methods can be used for each subsequent redirection.

The MASA service could lock a claim and refuse to issue a new token or the MASA service could go offline (for example if a vendor went out of business). This functionality provides benefits such as theft resistance, but it also implies an operational risk to the Domain that Vendor behavior could limit future bootstrapping of the device by the Domain. This can be mitigated by Registrars that request nonce-less authorization tokens.

8. Acknowledgements

We would like to thank the various reviewers for their input, in particular Markus Stenberg, Brian Carpenter, Fuyu Eleven.

9. References

9.1. Normative References

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

[RFC3542] Stevens, W., Thomas, M., Nordmark, E., and T. Jinmei, "Advanced Sockets Application Program Interface (API) for IPv6", RFC 3542, May 2003.

[RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, May 2005.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC5386] Williams, N. and M. Richardson, "Better-Than-Nothing Security: An Unauthenticated Mode of IPsec", RFC 5386, November 2008.

[RFC5660] Williams, N., "IPsec Channels: Connection Latching", RFC 5660, October 2009.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <http://www.rfc-editor.org/info/rfc6762>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <http://www.rfc-editor.org/info/rfc6763>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <http://www.rfc-editor.org/info/rfc7030>.

[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>.

9.2. Informative References

[I-D.behringer-homenet-trust-bootstrap]
Behringer, M., Pritikin, M., and S. Bjarnason, "Bootstrapping Trust on a Homenet", draft-behringer-homenet-trust-bootstrap-02 (work in progress), February 2014.

[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-03 (work in progress), March 2016.

[I-D.ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch-07 (work in progress), March 2016.

[I-D.irtf-nmrg-autonomic-network-definitions]
Behringer, M., Pritikin, M., Bjarnason, S., Clemm, A., Carpenter, B., Jiang, S., and L. Ciavaglia, "Autonomic Networking - Definitions and Design Goals", draft-irtf-nmrg-autonomic-network-definitions-07 (work in progress), March 2015.

[I-D.lear-mud-framework]
Lear, E., "Manufacturer Usage Description Framework", draft-lear-mud-framework-00 (work in progress), January 2016.

[I-D.richardson-anima-state-for-joinrouter]
Richardson, M., "Considerations for stateful vs stateless join router in ANIMA bootstrap", draft-richardson-anima-state-for-joinrouter-00 (work in progress), January 2016.

[imprinting]
Wikipedia, , "Wikipedia article: Imprinting", July 2015, <https://en.wikipedia.org/wiki/Imprinting_(psychology)>.

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

Authors' Addresses

Max Pritikin
Cisco

Email: pritikin@cisco.com

Michael C. Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z 5V7
CA

Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/

Michael H. Behringer
Cisco

Email: mbehring@cisco.com

Steinthor Bjarnason
Cisco

Email: sbjarnas@cisco.com