Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 21, 2019

O. Friel
E. Lear
J. Henry
Cisco
M. Richardson
Sandelman Software Works

October 18, 2018

BRSKI over IEEE 802.11
draft-friel-anima-brski-over-802dot11-00

Abstract

This document outlines the challenges associated with implementing Bootstrapping Remote Secure Key Infrastructures over IEEE 802.11 and IEEE 802.1x networks. Multiple options are presented for discovering and authenticating to the correct IEEE 802.11 SSID. This draft is a discussion document and no final recommendations are made on the recommended approaches to take. However, the advantages and downsides of each possible method are evaluated.

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 https://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 21, 2019.

Table of Contents

1. Introduction

Bootstrapping Remote Secure Key Infrastructures (BRSKI) [I-D.ietf-anima-bootstrapping-keyinfra] describes how a device can bootstrap against a local network using an Initial Device Identity X.509 [IEEE802.1AR] IDevID certificate that is pre-installed by the vendor on the device in order to obtain an [IEEE802.1AR] LDevID. The BRSKI flow assumes the device can obtain an IP address, and thus assumes the device has already connected to the local network. Further, the draft states that BRSKI use of IDevIDs:

> allows for alignment with [IEEE802.1X] network access control methods, its use here is for Pledge authentication rather than network access control. Integrating this protocol with network access control, perhaps as an Extensible Authentication Protocol (EAP) method (see [RFC3748], is out-of-scope.

The draft does not describe any mechanisms for how an [IEEE802.11] enabled device would discover and select a suitable [IEEE802.11] SSID when multiple SSIDs are available. A typical deployment scenario could involve a device begin deployed in a location were twenty or more SSIDs are being broadcast, for example, in a multi-tenanted building or campus where multiple independent organizations operate [IEEE802.11] networks.

In order to reduce the administrative overhead of installing new devices, it is desirable that the device will automatically discover and connect to the correct SSID without the installer having to manually provision any network information or credentials on the device. It is also desirable that the device does not discover, connect to, and automatically enroll with the wrong network as this could result in a device that is owned by one organization connecting to the network of a different organization in a multi-tenanted building or campus.

Additionally, as noted above, the BRSKI draft does not describe how BRSKI could potentially align with [IEEE802.1X] authentication mechanisms.

This document outlines multiple different potential mechanisms that would enable a bootstrapping device to choose between different available [IEEE802.11] SSIDs in order to associate and execute the BRSKI flow. This document also outlines several options for how [IEEE802.11] networks enforcing [IEEE802.1X] authentication could enable the BRSKI flow, and describes the required device behaviour.

This document presents both [IEEE802.11] mechanisms and Wi-Fi Alliance (WFA) mechanisms. An important consideration when determining what the most appropriate solution to device onboarding should be is what bodies need to be involved in standardisation efforts: IETF, IEEE and/or WFA.

1.1. Terminology

IEEE 802.11u: an amendment to the IEEE 802.11-2007 standard to add features that improve interworking with external networks.

ANI: Autonomic Networking Infrastructure

ANQP: Access Network Query Protocol

AP: IEEE 802.11 Access Point

CA: Certificate Authority

EAP: Extensible Authentication Protocol

EST: Enrollment over Secure Transport

HotSpot 2.0 / HS2.0: An element of the Wi-Fi Alliance Passpoint certificatoin program that enables cell phones to automatically discover capabilities and enroll into IEEE 802.11 guest networks (hotspots).

IE: Information Element

IDevID: Initial Device Identifier

LDevID: Locally Significant Device Identifier

OI: Organization Identifier

MASA: BRSKI Manufacturer Authorized Signing Authority service

SSID: IEEE 802.11 Service Set Identifier

STA: IEEE 802.11 station

WFA: Wi-Fi Alliance

WLC: Wireless LAN Controller

WPA/WPA2: Wi-Fi Protected Access / Wi-Fi Protected Access version 2

WPS: Wi-Fi Protected Setup

2. Discovery and Authentication Design Considerations

2.1. Incorrect SSID Discovery

As will be seen in the following sections, there are several discovery scenarios where the device can choose an incorrect SSID and attempt to join the wrong network. For example, the device is being deployed by one organization in a multi-tenant building, and chooses to connect to the SSID of a neighbor organization. The device is dependent upon either detecting that the other networks are unwanted candidates, or upon the incorrect networks rejecting its BRSKI enrollment attempt. It is possible that the device could end up enrolled with the wrong network. It is also possible that the device will waste time before identifying and joining the correct network.

2.1.1. Leveraging BRSKI MASA

2.1.1.1. Prevention

BRSKI allows optional sales channel integration which could be used to ensure only the "correct" network can claim the device. In theory, this could be achieved if the BRSKI MASA service has explicit knowledge of the network where every single device will be deployed. After connecting to the incorrect SSID and possibly authenticating to the network, the device would present network TLS information in its voucher-request, and the MASA server would have to reject the request based on this network TLS information and not issue a voucher. The device could then reject that SSID and attempt to bootstrap against the next available SSID.

This could possibly be acheieved via sales channel integration, where devices are tracked through the supply chain all the way from manufacturer factory to target deployment network operator. In practice, this approach may be challenging to deploy as it may be extremely difficult to implement this tightly coupled sales channel

integration and ensure that the MASA actually has accurate deployment network information.

An alternative to sales channel integration is to provide the device owners with a, possibly authenticated, interface or API to the MASA service whereby they would have to explicitly claim devices prior to the MASA issuing vouchers for that device. There are similar problems with this approach, as there could be a complex sales and channel partner chain between the MASA service operator and the device operator who owns and deploys the device. This could make exposure of APIs by the MASA operator to the device operator untenable.

2.1.1.2. Detection

If a device connects to the wrong network, the correct network operator could detect this incorrect association after the fact by integration with MASA and checking audit logs for the device. The MASA audit logs should indicate all networks that have been issued vouchers for a specific device. This mechanism also relies on the correct network operator having a list, bill or materials, or similar of all device identities that should be connecting to their network in order to check MASA logs for devices that have not come online, but are known to be physically deployed.

2.1.2. Relying on the Network Administrator

An obvious mechanism is to rely on network administrators to be good citizens and explicitly reject devices that attempt to bootstrap against the wrong network. This is not guaranteed to work for two main reasons:

- Some network administrators will configure an open policy on their network. Any device that attempts to connect to the network will be automatically granted access.

- Some network administrators will be bad actors and will accept the onboarding of devices that they do not own but that are in range of their networks.

2.1.3. Requiring the Network to Demonstrate Knowledge of Device

Technologies such as the WFA Easy Connect (also known as Device Provisioning Profile [DPP]) require that a network provisoining entity demonstrates knowledge of device information such as the device's bootstrapping public key prior to the device attempting to connect to the network. This gives a higher level of confidence to the device that it is connecting to the correct SSID. These

mechanisms could leverage a key that is printed on the device label, or included in a sales channel bill of materials. The security of these types of key distribution mechanisms relies on keeping the device label or bill of materials content from being compromised prior to device installation.

[IEEE802.11] also includes several advertisement mechanisms that could allow the device to exchange information with the wireless infrastructure. Examples are provided throughout this text. Such exchange can be added to, or integrated with, the standard [IEEE802.11] discovery mechanisms to allow the device to discard the networks that would not provide information showing that the network knows the device. Similarly, the network could reject the association of devices that would fail to show particular indicators related to their credentials.

2.2. IEEE 802.11 Authentication Mechanisms

[IEEE802.11i] allows an SSID to advertise different authentication mechanisms via the AKM Suite list in the RSNE. A very brief introduction to [IEEE802.11i] is given in the appendices. An SSID could advertise PSK or [IEEE802.1X] authentication mechanisms. When a network operator needs to enforce two different authentication mechanisms, one for pre-BRSKI devices and one for post-BRSKI devices, the operator has four options:

- configure two SSIDs with the same SSID string value, each one advertising a different authentication mechanism
- configure two different SSIDs, each with its own SSID string value, with each one advertising a different authentication mechanism
- configure a single SSID, advertising two different authentication mechansim in the RSNE
- configure a single SSID, advertising a general authentication mechanism in the RSNE, and particular additional authentication options in some other information element.

If devices have to be flexible enough to handle two of more of these options, then this adds complexity to the device firmware and internal state machines. Similarly, if network infrastructure (APs, WLCs, AAAs) potentially needs to support all options, then this adds complexity to network infrastructure configuration flexibility, software and state machines. Consideration must be given to the practicalities of implementation for both devices and network infrastructure when designing the final bootstrap mechanism and

aligning [IEEE802.11], [IEEE802.1X] and BRSKI protocol interactions. As such, a mechanism that allows for the coexistence of pre-BRSKI and post-BRSKI authentication on the same SSID is likely to be preferred.

2.2.1. Authentication Signaling Considerations

Devices should be flexible enough to handle potential options defined by any final draft. When discovering a pre-BRSKI SSID, the device should also discover the authentication mechanisms enforced by the SSID. If the device supports the authentication mechanism being advertised, then the device can connect to the SSID in order to initiate the BRSKI flow. For example, the device may support [IEEE802.1X] as a pre-BRSKI authentication mechanism, but may not support PSK as a pre-BRSKI authentication mechanism.

Once the device has completed the BRKSI flow and has obtained an LDevID, a mechanism is needed to tell the device which SSID to use for post-BRSKI network access. This may be the same SSID as the pre-BRSKI SSID, or another SSID. The decision in whether to onboard devices through the production SSID or use an onboarding and provisioning SSID that is different from the production SSID is dependent on individual organisation networking and security architectures. As such, the mechanism by which the post-BRSKI SSID is advertised to the device, if that SSID is different from the pre-BRSKI SSID, is out-of-scope of this version of this document.

2.2.2. IP Address Assignment Considerations

If a device has to perform two different authentications, one for pre-BRSKI and one for post-BRSKI, network policy will typically assign the device to different VLANs for these different stages, and may assign the device different IP addresses depending on which network segment the device is assigned to. This could be true even if a single SSID is used for both pre-BRSKI and post-BRSKI connections. Therefore, the bootstrapping device may need to completely reset its network connection and network software stack, and obtain a new IP address between pre-BRSKI and post-BRSKI connections.

2.3. Client and Server Implementations

When evaluating all possible SSID discovery mechanisms and authentication mechanisms outlined in this document, consideration must be given to the complexity of the required client and server implementation and state machines. Consideration must also be given to the network operator configuration complexity if multiple permutations and combinations of SSID discovery and network authentication mechanisms are possible.

3. Potential SSID Discovery and Validation Mechanisms

This section outlines multiple different mechanisms that could potentially be leveraged that would enable a bootstrapping device to choose between multiple different available [IEEE802.11] SSIDs. The discovery mechanism needs to include the following steps:

- A process for the bootstrapping device that has not completed the bootstrapping process, and that it is at a stage where such process is needed before further connection
- A process for the Wi-Fi infrastructure to signal that it can perform bootstrapping
- A process for the bootstrapping device and the infrastructure to validate each other request. This step includes, for the bootstrapping device, discriminating between two SSIDs in range. This step may also include, for the Wi-Fi infrastructure, validating the bootstrapping device's request (before accepting it).

The discovery options outlined in this document include:

- Well-known BRSKI SSID
- [IEEE802.11aq]
- [IEEE802.11] Vendor Specific Information Element
- Reusing Existing [IEEE802.11u] Elements
- [IEEE802.11u] Interworking Information - Internet
- Define New [IEEE802.11u] Extensions
- Wi-Fi Protected Setup
- Define and Advertise a BRSKI-specific AKM in RSNE
- Wi-Fi Device Provisioning Profile

These mechanisms are described in more detail in the following sections.

3.1. Well-known BRSKI SSID

A standardized naming convention for SSIDs offering BRSKI services is defined such as:

- BRSKI%ssidname

Where:

- BRSKI: is a well-known prefix string of characters. This prefix string would be baked into device firmware.
- %: is a well known delimiter character. This delimiter character would be baked into device firmware.
- ssidname: is the freeform SSID name that the network operator defines.

Device manufacturers would bake the well-known prefix string and character delimiter into device firmware. Network operators configuring SSIDs which offer BRSKI services would have to ensure that the SSID of those networks begins with this prefix. On bootstrap, the device would scan all available SSIDs and look for ones with this given prefix.

If multiple SSIDs are available with this prefix, then the device could simply round robin through these SSIDs and attempt to start the BRSKI flow on each one in turn until it succeeds.

This mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device enrolling against an incorrect network.

Another issue with defining a specific naming convention for the SSID is that this may require network operators to have to deploy a new SSID. In general, network operators attempt to keep the number of unique SSIDs deployed to a minimum as each deployed SSID eats up a percentage of available air time and network capacity. A good discussion of SSID overhead and an SSID overhead [calculator] is available.

Additionally, a third issue with this mechanism is that the bootstrapping SSID might be different from the production SSID. As such, using this mechanism may force a network operator to maintain an SSID (with the overhead concerns detailed above) just for occasional boostrapping events. The SSID could be enabled only when bootstrapping events are expected, but this manual operation does not scale very well (and ignores cases where devices need to re-bootstrap

or are introduced into the network individually at unpredictable intervals). Keeping the SSID enabled at all times consumes airtime for low added value outside of the bootstrapping events.

3.2. IEEE 802.11aq

[IEEE802.11aq] is an amedment to the [IEEE802.11] Standard that was published in August 2018. [IEEE802.11aq] defines new elements that can be included in [IEEE802.11] Beacon, Probe Request and Probe Response frames, and defines new elements for ANQP frames.

The extensions allow an AP to broadcast support for backend services, where allowed services are those registered in the [IANA] Service Name and Transport Protocol Port Number Registry. The services can be advertised in [IEEE802.11] elements that include either:

- SHA256 hashes of the registered service names
- a bloom filter of the SHA256 hashes of the registered service names

Bloom filters simply serve to reduce the size of Beacon and Probe Response frames when a large number of services are advertised. If a bloom filter is used by the AP, and a device discovers a potential service match in the bloom filter, then the device can query the AP for the full list of service name hashes using newly defined ANQP elements.

If BRSKI were to leverage [IEEE802.11aq], then a BRSKI service would need to be defined in [IANA].

[IEEE802.11aq] describes two types of exchanges. An unsollicited Preassociation Discovery (PAD) procedure, where the AP advertises services reachable through the AP, and a sollicited method, where the PAD is initated by the unassociated client attempting to discover a service offered through the AP and SSID. The unsollictited PAD method could be leveraged to advertise support for BRSKI. This mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device enrolling against an incorrect network.

The sollicited method could be used by the device to query about general BRSKI support, or to request information about specific BRSKI modes or options. This method could be used to overcome the Section 2.1 issue.

3.3. IEEE 802.11 Vendor Specific Information Element

[IEEE802.11] defines Information Element (IE) number 221 for carrying Vendor Specific information. The purpose of this document is to define an SSID discovery mechanism that can be used across all devices and vendors, so use of this IE is not an appropriate long term solution.

3.4. Reusing Existing IEEE 802.11u Elements

[IEEE802.11u] defines mechanisms for interworking. An introduction to [IEEE802.11u] is given in the appendices. Existing IEs in [IEEE802.11u] include:

- Roaming Consortium IE (RCOI)

- NAI Realm IE

These existing IEs could be used to advertise a well-known, logical service that devices implicitly know to look for. This may be implemented in the spirit of the 802.11u logic, where the NAI or the RCOI point to a specific set of service providers. This could also be implemented as a variation where the NAI or the RCOI point to a specific service, with no specific service provider identified in the IE.

In the case of NAI Realm, a well-known service name such as "_bootstrapks" could be defined and advertised in the NAI Realm IE. In the case of Roaming Consortium, a well-known Organization Identifier (OI) could be defined and advertised in the Roaming Consortium IE.

Device manufacturers would bake the well-known NAI Realm or Roaming Consortium OI into device firmware. Network operators configuring SSIDs which offer BRSKI services would have to ensure that the SSID offered this NAI Realm or OI. On bootstrap, the device would scan all available SSIDs and use ANQP to query for NAI Realms or Roaming Consortium OI looking for a match.

The key concept with this proposal is that BRSKI uses a well-known NAI Realm name or Roaming Consortium OI more as a logical service advertisement rather than as a backhaul internet provider advertisement. This is conceptually very similar to what [IEEE802.11aq] is attempting to achieve.

Leveraging NAI Realm or Roaming Consortium would not require any [IEEE802.11] specification changes, and could be defined by this IETF draft with the strings suggested above for NAI. However, the RCOI

has the format of a MAC address, and would need to be allocated by the IEEE. In the case where specific vendors would implement a specific NAI or RCOI, identifying both the vendor or vendor consortium and support for BRSKI, new NAI and RCOI would need to be defined by these vendors. Although the Wireless Broadband Alliance (WBA) keeps a Next generation Hotspot (NGH) registry of known RCOIs and NAIs, there is no official and exahsutive published repository of these values.

In addition to BRSKI support, as the NAI Realm includes advertising the EAP mechanism required, if a new EAP-BRSKI were to be defined, then this could be advertised. Devices could then scan for an NAI Realm that enforced EAP-BRSKI, and ignore the realm name.

This mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device enrolling against an incorrect network.

Additionally, as the IEEE is attempting to standardize logical service advertisement via [IEEE802.11aq], [IEEE802.11aq] would seem to be the more appropriate option than overloading an existing IE. However, it is worth noting that configuration of 802.11u IEs is commonly supported today by Wi-Fi infrastructure vendors, and this mechanism may be suitable for demonstrations or proof-of-concepts.

3.5. IEEE 802.11u Interworking Information - Internet

It is possible that an SSID may be configured to provide unrestricted and unauthenticated internet access. This could be advertised in the Interworking Information IE by including:

- internet bit = 1
- ASRA bit = 0

If such a network were discovered, a device could attempt to use the BRSKI well-known vendor cloud Registrar. Possibly this could be a default fall back mechanism that a device could use when determining which SSID to use. However, this mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device enrolling against an incorrect network. Additionally, this mechanism does not provide any information about local BRSKI support.

3.6. Define New IEEE 802.11u Extensions

Of the various elements currently defined by [IEEE802.11u] for potentially advertising BRSKI, NAI Realm and Roaming Consortium IE are the two existing options that are a closest fit, as outlined

above. Another possibility that has been suggested in the IETF mailers is defining an extension to [IEEE802.11u] specifically for advertising BRSKI service capability. Any extensions should be included in Beacon and Probe Response frames so that devices can discover BRSKI capability without the additional overhead of having to explicitly query using ANQP. ANQP queries could be used to provide additional information, such as vendor support.

[IEEE802.11aq] appears to be the proposed mechanism for generically advertising any service capability, provided that service is registered with [IANA]. It is probably a better approach to encourage adoption of [IEEE802.11aq] and register a service name for BRSKI with [IANA] rather than attempt to define a completely new BRSKI-specific [IEEE802.11u] extension.

3.7. Wi-Fi Protected Setup

Wi-Fi Protected Setup (WPS) only works with Wi-Fi Protected Access (WPA) and WPA2 when in Personal Mode. WPS does not work when the network is in Enterprise Mode enforcing [IEEE802.1X] authentication. WPS is intended for consumer networks and does not address the security requirements of enterprise or IoT deployments. Additionally, WPS relies on three methods (button push, PIN or NFC), none of which scale easily in an enterprise environement.

3.8. Define and Advertise a BRSKI-specific AKM in RSNE

[IEEE802.11i] introduced the RSNE element which allows an SSID to advertise multiple authentication mechanisms. A new Authentication and Key Management (AKM) Suite could be defined that indicates the STA can use BRSKI mechanisms to authenticate against the SSID. The authentication handshake could be an [IEEE802.1X] handshake, possibly leveraging an EAP-BRSKI mechanism, the key thing here is that a new AKM is defined and advertised to indicate the specific BRSKI-capable EAP method that is supported by [IEEE802.1X], as opposed to the current [IEEE802.1X] AKMs which give no indication of the supported EAP mechanisms. It is clear that such method would limit the SSID to BRSKI-supporting clients. This would require an additional SSID specifically for BRSKI clients. As such, this solution also suffers from the limitations mentioned about additional overhead. Additionally, this mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device attempting to enroll against an incorrect network.

3.9. Wi-Fi Device Provisioning Profile

The [DPP] specification, also known as Wi-Fi Easy Connect, defines how an entity that is already trusted by a network can assist an untrusted entity in enrolling with the network. The description below assumes the [IEEE802.11] network is in infrastructure mode. DPP introduces multiple key roles including:

- Configurator: A logical entity that is already trusted by the network that has capabilities to enroll and provision devices called Enrollees. A Configurator may be a STA or an AP.

- Enrollee: A logical entity that is being provisioned by a Configurator. An Enrollee may be a STA or an AP.

- Initiator: A logical entity that initiates the DPP Authentication Protocol. The Initiator may be the Configurator or the Enrollee.

- Responder: A logical entity that responds to the Initiator of the DPP Authentication Protocol. The Responder may be the Configurator or the Enrollee.

In the DPP model, a common Configurator and Initiator is an app running on a trusted smartphone. This process is manual, and each device is treated individually. In order to support a plug and play model for installation of a large number devices, where each device is simply powered up for the first time and automatically discovers the Wi-Fi network without the need for a helper or supervising application, then this implies that the AP must perform the role of the Configurator and the device or STA performs the role of Enrollee. Note that the AP may simply proxy DPP messages through to a backend WLC, but from the perspective of the device, the AP is the Configurator.

The DPP specification also mandates that the Initiator must be bootstrapped the bootstrapping public key of the Responder. For BRSKI purposes, the DPP bootstrapping public key will be the [IEEE802.1AR] IDevID of the device. As the boostrapping device cannot know in advance the bootstrapping public key of a specific operators network, this implies that the Configurator must take on the role of the Initiator. Therefore, the AP must take on the roles of both the Configurator and the Initiator.

At boot time, the device does not know which AP or which SSID is likely to provide DPP services. In the DPP model, the Configurator advertizes a special Authentication and Key Management (AKM) mode, DPP. Announcing this mode outside of onboarding windows might result in regular, non-DPP clients to fail to associate to a network which

AKM they do not recognize. As such, it is preferable that the DPP process be started after the device establishes a link with the access point. Therefore, DPP is likely not the best process to identify a supporting access point. Additionally, this mechanism suffers from the limitations outlined in Section 2.1 - it does nothing to prevent a device attempting to enroll against an incorrect network.

4. Potential Mutual Validation Options

When the bootstrapping device determines that one or more APs or SSIDs are available that provide support for BRSKI, with one or more of the mechanisms listed in section 3, then the device needs to determine which is the correct SSID. At the same time, an AP receiving signals from a bootstrapping device may need to verify if the need to determine if the device is attempting to connect the the correct network. In essence, this joint requirement means that BRSKi could be started immediately after the discovery phase. A case of mistaken identity (device attempting to join the wrong network) can be resolved with a round robin process, where the device fails the BRSKI process on the attempted network, then attempts BRSKi against the next candidate network. However, this process may result in wasted airtime and possible security exposure where an operator attempts to capture information about neighboring bootstrapping devices.

4.1. MAC Address Validation method

An alternative to the round robin mode is a primary selection mode where the device and the AP exchange mutual signs of knowledge about each other. This could be achieved using the standard 802.11 process, where the device would send a probe request using its real MAC address. This MAC address could be known to a central database and validated by the wireless infrastructure. This method has the merit of being simple. However, it is more and more common for devices with simple network stacks to use locally administered (and temporal) MAC addresses. This method only validates the device (not the infrastructure).

4.2. Vendor Token Validation method

An alternative to the MAC address method is to use a token, placed in an extension Information Element of the device probe request frame. This token would identify the device vendor. A limitation of this method is that, in some cases, neighboring networks may bootstrap devices from the same vendor. This method validates the vendor, but not the device. It also does not validate the infrastructure. It can be used as a coarse initial filtering mechanism.

4.3. Device Token Validation method

An alternative to the vendor token is to use a unique identifier for the device. However, as the transaction is exposed to eavesdropping, this method exposes the toke. As such, the token should not be an element that can be compromised. The token can be the MAC address, if the device uses locally administered addresses for its probe requests. This method only validates the device (not the infrastructure).

4.4. Infrastructure Response Filtering

When additional filtering is required, the infrastructure may validate the additiomal information provided by the device, and either respond, if the additional information is computed to match the infrastructure knowledge, or ignore the request (no probe response) if the additional information does not match the infrastructure knowledge.

In some cases, the AP may not be able to access the database locally, and may need to forward the request (including the additional information provided by the device) to another system. In this case, the AP may respond with a frame that includes a GAS comeback value. This value indicates a delay after which the device should ask the question again. In that interval, the AP will query the infrastructure to obtain the additional iformation required. After expiration of the comeback interval, the device may send the probe request again, and the AP may respond or ignore the request, or request more time. It is understood that the device would accept a limited number of comeback requests (for example 3) and a limited comeback interval (for example no more than 3 seconds).

4.5. Infrastructure Validation Method

It is expected, when the device adds information to its probe request, that the infrastructure should only respond to those devices that have been validated by the infrastructure system. However, some systems may not be able to respond in time and may be configured to accept all requests. Additionally, bad actors may decide to accept any request. There may therefore be a need to mandate the infrastructure to return information that indicates proof of knowledge of the device. The following modes are envisioned:

- When the device uses its MAC address, or expresses its MAC address in an information element contained in the probe request, the infrastructure may be able to express its knowledge of the device servial number, and mention this serial number in the probe

response. As it may be needed to protect the serial number at this stage, the serial number could be encoded in a bloom filter.

- When the device uses a vendor token, the AP can only reply with another token identifying the same vendor, as the device itself is not known.

5. Potential Authentication Options

When the bootstrapping device determines which SSID to connect to, there are multiple potential options available for how the device authenticates with the network while bootstrapping. Several options are outlined in this section. This list is not exhaustive.

At a high level, authentication can generally be split into two phases using two different credentials:

- Pre-BRSKI: The device can use its [IEEE802.1AR] IDevID to connect to the network while executing the BRSKI flow

- Post-BRSKI: The device can use its [IEEE802.1AR] LDevID to connect to the network after completing BRSKI enrollment

The authentication options outlined in this document include:

- Unauthenticated Pre-BRSKI and EAP-TLS Post-BRSKI

- DPP Pre-BRSKI and EAP-TLS Post-BRSKI

- PSK or SAE Pre-BRSKI and EAP-TLS Post-BRSKI

- MAC Address Bypass Pre-BRSKI and EAP-TLS Post-BRSKI

- EAP-TLS Pre-BRSKI and EAP-TLS Post-BRSKI

- New DPP BRSKI mechanism

- New TEAP BRSKI mechanism

- New [IEEE802.11] Authentication Algorithm for BRSKI and EAP-TLS Post-BRSKI

- New [IEEE802.1X] EAPOL-Announcements to encapsulate BRSKI prior to EAP-TLS Post-BRSKI

These mechanisms are described in more detail in the following sections. Note that any mechanisms leveraging [IEEE802.1X] are

[IEEE802.11] MAC layer authentication mechanisms and therefore the SSID must advertise WPA2 capability.

When evaluating the multiple authentication options outlined below, care and consideration must be given to the complexity of the software state machine required in both devices and services for implementation.

5.1. Unauthenticated and Unencrypted or OWE Pre-BRSKI and EAP-TLS Post-BRSKI

The device connects to an unauthenticated network pre-BRSKI. The device connects to a network enforcing EAP-TLS post-BRSKI. The device uses its LDevID as the post-BRSKI EAP-TLS credential.

In the pre-BRSKI phase, the device may establish a secure connection with the AP using WPA3 to protect the BRSKI exchange from eavesdroppers. The pre-BRSKi phase can be protected, but is not authenticated.

5.2. DPP Pre-BRSKI and EAP-TLS post-BRSKI

The device can be provisioned with DPP for the pre-BRSKI phase, receiving the SSID value and optionally a temporal PSK. It should be noted that the device at that point is not untampered anymore. However, the configuration is temporal and limited. In a WPA3 network, when DPP from a mobile (e.g. smartphone) is used, the DPP process may provision the SSID and leave the device to use OWE for its connection to the AP.

Alternatively, when DPP is processed through the AP in an automated fashion, the AP first establishes an OWE connection with the device. Through this encrypted connection, the AP provides the SSID and the temporal PSK value.

5.3. PSK or SAE Pre-BRSKI and EAP-TLS Post-BRSKI

The device connects to a network enforcing PSK pre-BRSKI. If DPP is not used, the PSK may be factory-set (default PSK) or provisioned by direct action on the device. Neither of these modes is preferred as factory-defauls are weak and direct interaction with the device does not allow for massive automated bootstrapping. After the PSK-based pre-BRSKI connection, the device connects to a network enforcing EAP-TLS post-BRSKI. The device uses the LDevID obtained via BRSKI as the post-BRSKI EAP-TLS credential.

When the device connects to the post-BRSKI network that is enforcing EAP-TLS, the device uses its LDevID as its credential. The device

should verify the certificate presented by the server during that EAP-TLS exchange against the trusted CA list it obtained during BRSKI.

If the [IEEE802.1X] network enforces a tunneled EAP method, for example [RFC7170], where the device must present an additional credential such as a password, the mechanism by which that additional credential is provisioned on the device for post-BRSKI authentication is out-of-scope of this version of this document. NAI Realm may be used to advertise the EAP methods being enforced by an SSID. It is to be determined if guidelines should be provided on use of NAI Realm for advertising EAP method in order to streamline BRSKI.

5.4. MAC Address Bypass Pre-BRSKI and EAP-TLS Post-BRSKI

Many AAA server state machine logic allows for the network to fallback to MAC Address Bypass (MAB) when initial authentication against the network fails. If the device does not present a valid credential to the network, then the network will check if the device's MAC address is whitelisted. If it is, then the network may grant the device access to a network segment that will allow it to complete the BRSKI flow and get provisioned with an LDevID. Once the device has an LDevID, it can then reauthenticate against the network using its EAP-TLS and its LDevID.

5.5. EAP-TLS Pre-BRSKI and EAP-TLS Post-BRSKI

The device connects to a network enforcing EAP-TLS pre-BRSKI. The device uses its IDevID as the pre-BRSKI EAP-TLS credential. The device connects to a network enforcing EAP-TLS post-BRSKI. The device uses its LDevID as the post-BRSKI EAP-TLS credential.

When the device connects to a pre-BRSKI network that is enforcing EAP-TLS, the device uses its IDevID as its credential. The device should not attempt to verify the certificate presented by the server during that EAP-TLS exchange, as it has not yet discovered the local domain trusted CA list.

When the device connects to the post-BRSKI network that is enforcing EAP-TLS, the device uses its LDevID as its credential. The device should verify the certificate presented by the server during that EAP-TLS exchange against the trusted CA list it obtained during BRSKI.

Again, if the post-BRSKI network enforces a tunneled EAP method, the mechanism by which that second credential is provisioned on the device is out-of-scope of this version of this document.

5.6. New DPP BRSKI mechanism

BRSKI can be integrated into the DPP choreography, in three modes:

- When a local commissioning tool is used (e.g. application on a mobile device), the standard DPP process is used for the configurator to establish a trusted connection to the enrolee (the bootstrapping device), over Bluetoot, NFC, Wi-Fi or other means defined by DPP. The configurator then provision the boostrapping device with the target SSID, but also installs on the device the TrustAnchor. The bootstrapping device then connects to the target SSID using EAP-BRSKI (EST). The query is relayed to the registrar, which validates the device identity. An EAP-Success message is then returned to the access point.

- When the commissioning tool is not mobile and not interacting directly with the bootstrapping device, identifiers for the device may be fed into an authentication database (e.g. serial number, MAC address, DPP key, device-specific factory-set PSK or other). Upon device request (probe request with request for network proof of knowledge), the AP retrieves one or more of these parameters from the authentication database, and uses them to provide proof of knowledge to the device. Once trust is established, a temporal trusted link is established between the device and the AP (using DPP parameters or OWE) and the AP provisions the device with the SSID. The device then connects to the target SSID using EAP-BRSKI as above.

- When the authentication server has reachability to the MASA server, the process above is started. As the device conencst to the target SSID, its identity is not only validated by the authentication server, but the authentication server also initiates a voucher request to the MASA server. The exchange between the bootstrapping device and the authentication server, now in possession of the voucher, continues as per [I-D.ietf-anima-bootstrapping-keyinfra].

5.7. New TEAP BRSKI mechanism

New TEAP TLVs are defined to transport BRSKI messages inside an outer EAP TLS tunnel such as TEAP [RFC7170]. [I-D.lear-eap-teap-brski] outlines a proposal for how BRSKI messages could be transported inside TEAP TLVs. At a high level, this enables the device to obtain an LDevID during the Layer 2 authentication stage. This has multiple advantages including:

- avoids the need for the device to potentially connect to two different SSIDs during bootstrap

- the device only needs to handle one authentication mechanism during bootstrap
- the device only needs to obtain one IP address, which it obtains after BRSKI is complete
- avoids the need for the device to have to disconnect from the network, reset its network stack, and reconnect to the network
- potentially simplifies network policy configuration

There are two suboptions to choose from when tunneling BRSKI messages inside TEAP:

- define new TLVs for transporting BRSKI messages inside the TEAP tunnel
- define a new EAP BRSKI method type that is tunneled within the outer TEAP method

This section assumes that new TLVs are defined for transporting BRSKI messages inside the TEAP tunnel and that a new EAP BRSKI method type is not defined.

The device discovers and connects to a network enforcing TEAP. A high level TEAP with BRSKI extensions flow would look something like:

- Device starts the EAP flow by sending the EAP TLS ClientHello message
- EAP server replies and includes CertificateRequest message, and may specify certificate_authorities in the message
- if the device has an LDevID and the LDevID issuing CA is allowed by the certificate_authorities list (i.e. the issuing CA is explicitly included in the list, or else the list is empty) then the device uses its LDevID to establish the TLS tunnel
- if the device does not have an LDevID, or certificate_authorities prevents it using its LDevID, then the device uses its IDevID to establish the TLS tunnel
- if certificate_authorities prevents the device from using its IDevID (and its LDevID if it has one) then the device fails to connect

The EAP server continues with TLS tunnel establishment:

- if the device certificate is invalid or expired, then the EAP server fails the connection request.

- if the device certificate is valid but is not allowed due to a configured policy on the EAP server, then the EAP server fails the connection request

- if the device certificate is accepted, then the EAP server establishes the TLS tunnel and starts the tunneled EAP-BRSKI procedures

At this stage, the EAP server has some policy decisions to make:

- if network policy indicates that the device certificate is sufficient to grant network access, whether it is an LDevID or an IDevID, then the EAP server simply initiates the Crypto-Binding TLV and 'Success' Result TLV exchange. The device can now obtain an IP address and connect to the network.

- the EAP server may instruct the device to initialise a full BRSKI flow. Typically, the EAP server will instruct the device to initialize a BRSKI flow when it presents an IDevID, however, the EAP server may instruct the device to initialize a BRSKI flow even if it presented a valid LDevID. The device sends all BRSKI messages, for example 'requestvoucher', inside the TLS tunnel using new TEAP TLVs. Assuming the BRSKI flow completes successfully and the device is issued an LDevID, the EAP server completes the exchange by initiating the Crypto-Binding TLV and 'Success' Result TLV exchange.

Once the EAP flow has successfully completed, then:

- network policy will automatically assign the device to the correct network segment

- the device obtains an IP address

- the device can access production service

It is assumed that the device will automatically handle LDevID certificate reenrolment via standard EST [RFC7030] outside the context of the EAP tunnel.

An item to be considered here is what information is included in Beacon or Probe Response frames to explicitly indicate that [IEEE802.1X] authentication using TEAP supporting BRSKI extensions is allowed. Currently, the RSNE included in Beacon and Probe Response frames can only indicate [IEEE802.1X] support.

5.8. New IEEE 802.11 Authentication Algorithm for BRSKI and EAP-TLS Post-BRSKI

[IEEE802.11] supports multiple authentication algorithms in its Authentication frame including:

- Open System
- Shared Key
- Fast BSS Transition
- Simultaneous Authentication of Equals

Shared Key authentication is used to indicate that the legacy WEP authentication mechanism is to be used. Simultaneous Authentication of Equals is used to indicate that the Dragonfly-based shared passphrase authentication mechanism introduced in [IEEE802.11s] is to be used. One thing that these two methods have in common is that a series of handshake data exchanges occur between the device and the AP as elements inside Authentication frames, and these Authentication exchanges happen prior to [IEEE802.11] Association.

It would be possible to define a new Authentication Algorithm and define new elements to encapsulate BRSKI messages inside Authentication frames. For example, new elements could be defined to encapsulate BRSKI requestvoucher, voucher and voucher telemetry JSON messages. The full BRSKI flow completes and the device gets issued an LDevID prior to associating with an SSID, and prior to doing full [IEEE802.1X] authentication using its LDevID.

The high level flow would be something like:

- SSID Beacon / Probe Response indicates in RSNE that it supports BRSKI based Authentication Algorithm
- SSIDs could also advertise that they support both BRSKI based Authentication and [IEEE802.1X]
- device discovers SSID via suitable mechanism
- device completes BRSKI by sending new elements inside Authentication frames and obtains an LDevID
- device associates with the AP
- device completes [IEEE802.1X] authentication using its LDevID as credential for EAP-TLS or TEAP

5.9. New IEEE 802.1X EAPOL-Announcements to encapsulate BRSKI and EAP-TLS Post-BRSKI

[IEEE802.1X] defines multiple EAPOL packet types, including EAPOL-Announcement and EAPOL-Announcement-Req messages. EAPOL-Annoncement and EAPOL-Announcement-Req messages can include multiple TLVs. EAPOL-Annoncement messages can be sent prior to starting any EAP authentication flow. New TLVs could be defined to encapsulate BRSKI messages inside EAPOL-Announcement and EAPOL-Announcement-Req TLVs. For example, new TLVs could be defined to encapsulate BRSKI requestvoucher, voucher and voucher telemetry JSON messages. The full BRSKI flow could complete inside EAPOL-Announcement exchanges prior to sending EAPOL-Start or EAPOL-EAP messages.

The high level flow would be something like:

- SSID Beacon / Probe Response indicates somehow in RSNE that it supports [IEEE802.1X] including BRSKI extensions.
- device connects to SSID and completes standard Open System Authentication and Association
- device starts [IEEE802.1X] EAPOL flow and uses new EAPOL-Announcement frames to encapsulate and complete BRSKI flow to obtain an LDevID
- device completes [IEEE802.1X] authentication using its LDevID as credential for EAP-TLS or TEAP

6. IANA Considerations

This document has no IANA actions.

7. Security Considerations

The mechanisms described in this document rely on BRSKI. As such, the same security considerations are applicable to this document as they are in [I-D.ietf-anima-bootstrapping-keyinfra].

Additionally, the Wireless LAN presents a unique DOS attack vector, as endpoints contend for the shared medium on a completely egalitarian basis with the AP. This means that any wireless device could potentially monopolize the air by constantly sending frames. This would prevent the bootstrapping device, or the infrastrcuture, to complete their exchange and would make the BRSKI process fail. This risk is inherent to the nature of 802.11 transmissions, and can only be mitigated by physical access control to the cell area. Such attack is also easily detected.

Also, initial exchanges between the bootstrapping device and the AP are not protected. Whenever a unicast communication is initiated between a bootstrapping device and an AP in an attempt to start active bootstrapping or provisioning, the link should first be protected whenever possible, for example with OWE.

7.1. Client side exposure

The discovery mechanism imposes that the bootstrapping device and the infrastructure must exchange messages to be aware of each other's existence. If these messages are generic, then the bootstrapping device has no mechanism to distinguish the correct SSID from a neighboring SSID. The bootstrapping device then is faced with two options:

- Try all possible SSIDs in a round-robin fashion. By doing so, the bootstrapping device will potentially expose parameters to the wrong SSID and infrastructure. Although such exposure is unlikely tor esult in device compromission, it will still expose unnecessarily device parameters to the wrong network. As such, it is recommended that a pre-BRSKI filtering mechanism be implemented to avoid this exposure, conducting the bootstrapping device to only start the BRSKI process with an SSID that has been confirmed to be a likely correct candidate.

- When the boostrapping device attempts to proceed to an SSID filtering, it may need to expose parameters to allow for the infrastructure to respond and provide a proof of knowledge. If this mechanism is implemented, the bootstrapping device should only expose information that is not sufficient to acquire complete knwledge of the bootstrapping device. For example, the bootstrapping device should not send both its serial number and MAC address, but should only expose an element that has low security value (such as a MAC address), and only in scenarios where the infrastructure has to respond with another element that will confirm to the bootstrapping device that it is communicating with the correct infrastructure.

7.2. Infrastructure side exposure

The general choreography of 802.11 networks imply that the infrastructure advertizes capabilities and support for specific features through beacons and probe responses. As such, the AP is likely to have to expose its support for BRSKI. This exposure is not a security concern.

When the infrastructure is requested to provide bre-BRSKI proof of knowledge, it has to process a frame received from an unknown

candidate device and either respond (if the device is found to be known), delay the response (if additional processing is needed) or ignore the request. Each of these behaviors may be tested by a rogue device in an attempt to gain information about the wireless infrastructure. It is therefore recommended that the proof of knowledge test should only focus on parameters specific to a particular device, and not to parameters generally applicable to multiple devices (for example parameters that would apply to multiple devices of one or more vendors).

8. Informative References

[calculator]
Revolution Wi-Fi, "SSID Overhead Calculator", n.d., <http://www.revolutionwifi.net/revolutionwifi/p/ssid-overhead-calculator.html>.

[DPP] Wi-Fi Alliance, "Wi-Fi Device Provisioning Protocol", n.d., <https://www.wi-fi.org/file/wi-fi-device-provisioning-protocol-dpp-draft-technical-specification-v0023>.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-16 (work in progress), June 2018.

[I-D.lear-eap-teap-brski]
Lear, E., Friel, O., and N. Cam-Winget, "Bootstrapping Key Infrastructure over EAP", draft-lear-eap-teap-brski-01 (work in progress), October 2018.

[IANA] Internet Assigned Numbers Authority, "Service Name and Transport Protocol Port Number Registry", n.d., <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

[IEEE802.11]
IEEE, ., "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", 2016.

[IEEE802.11aq]
IEEE, ., "802.11 Amendment 5 Pre-Association Discovery", 2017.

[IEEE802.11i]
IEEE, ., "802.11 Amendment 6 Medium Access Control (MAC) Security Enhancements", 2004.

[IEEE802.11s]
IEEE, ., "802.11 Amendment 10 Mesh Networking", 2011.

[IEEE802.11u]
IEEE, ., "802.11 Amendment 9 Interworking with External Networks", 2011.

[IEEE802.1AR]
IEEE, ., "Secure Device Identity", 2017.

[IEEE802.1X]
IEEE, ., "Port-Based Network Access Control", 2010.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, DOI 10.17487/RFC4282, December 2005, <https://www.rfc-editor.org/info/rfc4282>.

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

[RFC7170] Zhou, H., Cam-Winget, N., Salowey, J., and S. Hanna, "Tunnel Extensible Authentication Protocol (TEAP) Version 1", RFC 7170, DOI 10.17487/RFC7170, May 2014, <https://www.rfc-editor.org/info/rfc7170>.

Appendix A. IEEE 802.11 Primer

A.1. IEEE 802.11i

802.11i-2004 is an IEEE standard from 2004 that improves connection security. 802.11i-2004 is incorporated into 802.11-2014. 802.11i defines the Robust Security Network IE which includes information on:

- Pairwise Cipher Suites (WEP-40, WEP-104, CCMP-128, etc.)

- Authentication and Key Management Suites (PSK, 802.1X, etc.)

The RSN IEs are included in Beacon and Probe Response frames. STAs can use this frame to determine the authentication mechanisms offered by a particular AP e.g. PSK or 802.1X.

A.2. IEEE 802.11u

802.11u-2011 is an IEEE standard from 2011 that adds features that improve interworking with external networks. 802.11u-2011 is incorporated into 802.11-2016.

STAs and APs advertise support for 802.11u by setting the Interworking bit in the Extended Capabilities IE, and by including the Interworking IE in Beacon, Probe Request and Probe Response frames.

The Interworking IE includes information on:

- Access Network Type (Private, Free public, Chargeable public, etc.)
- Internet bit (yes/no)
- ASRA (Additional Step required for Access - e.g. Acceptance of terms and conditions, On-line enrollment, etc.)

802.11u introduced Access Network Query Protocol (ANQP) which enables STAs to query APs for information not present in Beacons/Probe Responses.

ANQP defines these key IEs for enabling the STA to determine which network to connect to:

- Roaming consortium IE: includes the Organization Identifier(s) of the roaming consortium(s). The OI is typically provisioned on cell phones by the SP, so the cell phone can automatically detect 802.11 networks that provide access to its SP's consortium.
- 3GPP Cellular Network IE: includes the Mobile Country Code (MCC) and Mobile Network Code (MNC) of the SP the AP provides access to.
- Network Access Identifier Realm IE: includes [RFC4282] realm names that the AP provides access to (e.g. wifi.service-provider.com). The NAI Realm IE also includes info on the EAP type required to access that realm e.g. EAP-TLS.
- Domain name IE: the domain name(s) of the local AP operator. Its purpose is to enable a STA to connect to a domain operator that may have a roaming agreement with STA's Service Provider.

STAs can use one or more of the above IEs to make a suitable decision on which SSID to pick.

HotSpot 2.0 is an example of a specification built on top of 802.11u and defines 10 additional ANQP elements using the standard vendor extensions mechanisms defined in 802.11. It also defines a HS2.0 Indication element that is included in Beacons and Probe Responses so that STAs can immediately tell if an SSID supports HS2.0.

Authors' Addresses

Owen Friel
Cisco

Email: ofriel@cisco.com

Eliot Lear
Cisco

Email: lear@cisco.com

Jerome Henry
Cisco

Email: jerhenry@cisco.com

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca