Network Working Group M.
Sethi
Internet-Draft
Ericsson
Intended status: Informational October 18,
2018
Expires: April 21, 2019

Enabling Network Access for IoT devices from the Cloud
draft-st-t2trg-nw-access-01

Abstract

This document describes a method for enabling and configuring network access for IoT devices that are first authenticated at a server. This server may be run by the manufacturer of the IoT device as an online cloud service. This specification is intended for off-the-shelf IoT devices that have just been purchased by the user. Many of these devices have only limited user interfaces that can be used for configuring network access credentials. The device configuration is also made more challenging by the fact that these devices may exist in large numbers.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

There is an increase in the deployment of Internet of Things (IoT) appliances such as wireless baby monitors, printers, speakers and smart TVs. To enable rapid adoption while reducing the cost of deployment, these appliances typically use the existing Wi-Fi infrastructure (Access Point) for Internet connectivity. However, configuring the network-access credentials for these off-the-shelf appliances is cumbersome. Typically this process requires the user to pair the appliance with his/her smartphone over bluetooth and then configure the Wi-Fi SSID and passphrase.

This process is not only cumbersome, but requires the appliance to be shipped with an additional network interface (only for configuration). It also moves the problem of securely configuring the network-access credentials to the problem of secure bluetooth pairing. Besides, relying on a single passphrase for the entire network may not be sustainable in the long run. While changing the passphrase to revoke/remove a device from the network is easy today when most devices have a keyboard and only a few (2-5) devices are connected to the network (Access Point), this would be much harder when the devices are many (10-100) and have limited input capabilities.

Once configured and connected to the Internet, the user still has to register the IoT device with the manufacturer. This maybe to receive services or software updates. For example, the user may connect his/her Wi-Fi weighing scale to keep track his/her weight online and receive software updates for new features.

This draft explains an example deployment scenario that relies on 802.1x [IEEE-802.1X] and EAP [RFC3748] authentication to register the device with the manufacturer and enable network access (provision WiFi credentials) for the IoT device at the same time. Using the 802.1x authentication even in SOHO (small office and home) scenarios is a big assumption. The following arguments may correctly apply against such a model:

- o Most home access points currently do not support 802.1x authentication: This is however in most cases only a software limitation. Many existing APs can support 802.1x authentication after firmware updates.

- o Home users do not understand RADIUS [RFC6929] peering and cannot configure 802.1x authentication: This is often very true. However, there are mechanisms with which the burden on the user can be significantly reduced. We will discuss some possible alternatives in the next section.

- o Most SOHO (Small office and Home) deployments are small and a network wide shared secret provides reasonable security: This is an incorrect assumption. While the deployments are small today, as more and more physical devices such as barbie dolls [barbie], weighing scales [scale], door bells [doorbell], and thermostats [nest] are connected to the Internet, using the same secret for the entire network is no longer sustainable. This is necessary to prevent attacks where for example, a compromised WiFi weighing scale also compromises the NEST thermostat that is using the same network secret.

- o An enterprise simply won't trust an external entity to remotely control their network and add new devices: This is true. However, what we are suggesting in this memo is to allow an IoT device manufacturer to put a new IoT device into a separate Virtual LAN and enable limited Internet connectivity for it. It is possible that certain enterprises may be willing. However, we accept that this may not be the case in all enterprise settings.

The architecture and solution presented in this draft is a generalized version of the original idea presented by Sethi et al. [Sethi14].

2. Terminology

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

3. Deployment Architecture

We first look at deployments where the online cloud service is run by the manufacturer. One such deployment architecture is shown in Figure 1. The IoT device is shown on the left. The device uses EAP for network-access authentication.

The Manufacturer IoT device registration portal is shown on the right. The manufacturer is responsible for running a AAA server that authenticates the IoT devices and then informs the users Access Point (AP) to enable Internet connectivity for the IoT device.

The Access Point (AP) at the user premises is shown in the middle. The AP provides Internet connectivity to the user devices. The AP must support 802.1x authentication and it uses RADIUS [RFC6929] or DIAMETER [RFC6733] to communicate with the Manufacturer IoT device registration portal. For simplicity, the rest of this memo uses RADIUS as the example protocol. However, it should be noted that the same objectives can be achieved with DIAMETER.

As shown in Figure 1 the AP may optionally have a local RADIUS server (which maybe the case in small office environments). In another deployment scenario shown in Figure 2, it is possible that the AP only has a local RADIUS client and routes all the EAP authentication messages to a single online RADIUS server (which maybe the case in home environments). In this case (Figure 2), the online RADIUS server may be run by the AP manufacturer for example. This would unburden the user from the task of maintaining a secure RADIUS server and setting up the necessary RADIUS peerings for IoT devices from different manufacturers.

For routing the EAP messages between the IoT device and the manufacturer portal, a RADIUS peering is needed between the AP (authenticator) and the AAA server that is run by the manufacturer. This peering may be secured with a shared secret or certificate-based TLS.

For correct routing of EAP messages from an IoT device to the device portal of the manufacturer, the realm part of the Network Access Identifier (NAI) [RFC7542] is used by the local RADIUS server in the AP (Figure 1) or the online RADIUS server (Figure 2). For example, the RADIUS server in either case could see that the NAI is of the form "device@examplevendor.com" and proxy the authentication request to the online service run by the Example Vendor at aaa.examplevendor.com on port 1812. As stated, the vendor service would only allow authentication request from trusted RADIUS servers that have peering relationship. The vendor service will then run several rounds of EAP message exchanges to authenticate the device.

On successful authentication, the vendor service informs the RADIUS server to enable network access for the device by sending a RADIUS Access-Accept message.

```
                                               +------------------+
                                               |Manufacturer IoT  |
                        +------------+         |  device portal   |
                        |Access Point|         | +----------+     |
   +-----------+        | +--------+ | RADIUS  | |  RADIUS  |     |
   |           |        | | RADIUS +---------------+ Server  |     |
   |   IoT     | EAP    | | Server | |DIAMETER | +-----+----+     |
   |  Device   +-------+ +--------+ |         |       |          |
   |           |        |            |         | +-----+----+     |
   +-----------+        |            |         | |    AAA   |     |
                        +------------+         | |   Server |     |
                                               | +----------+     |
                                               +------------------+
```

Figure 1: Deployment Architecture (Small Office)

```
                                                +------------------+
                                                |Manufacturer IoT  |
                    +------------+  +-----------+ |device portal     |
                    |Access Point|  |AP Manf.   | | +----------+     |
   +-------+        | +--------+ |  |Service    | | |RADIUS    |     |
   | IoT   |        | | RADIUS | |  | +-------+------+Server   |     |
   | Device| EAP| | Client +------+RADIUS | | | +-----+----+     |
   |       +----+ +--------+ |  | |Server | | |       |          |
   +-------+        |            |  | +-------+ | | +-----+----+     |
                    |            |  |           | | |    AAA   |     |
                    +------------+  +-----------+ | |   Server |     |
                                                | +----------+     |
                                                +------------------+
```

Figure 2: Deployment Architecture (Home)

The exact EAP method used for authentication can be decided by the IoT device manufacturer. For example, the manufacturer may
provision
certificates on the device which are then used for EAP-TLS [RFC5216] authentication. After successful authentication, the AAA server sends a RADIUS Access-Accept message enabling Internet connectivity for the IoT device. An example message flow in shown in Figure 3.

```
           IoT device             AP                 Manufacturer IoT
                                                      device portal
      -------------------     -------------
--------------------
                              <- EAP-Request/
                              Identity
      EAP-Response/
      Identity (MyID) ->

                              Forward auth messages
                              to IoT device portal
                              by looking at realm
                              in MyID

                                                       <- EAP-Request/
                                                      EAP-Type=EAP-TLS
                                                        (TLS Start)
      EAP-Response/
      EAP-Type=EAP-TLS
      (TLS client_hello)->
                                                    <- EAP-Request/
                                                    EAP-Type=EAP-TLS
                                                    (TLS server_hello,
                                                      TLS certificate,
                                             [TLS server_key_exchange,]
                                              TLS certificate_request,
                                                 TLS server_hello_done)
      EAP-Response/
      EAP-Type=EAP-TLS
      (TLS certificate,
       TLS client_key_exchange,
       TLS certificate_verify,
       TLS change_cipher_spec,
       TLS finished) ->
                                                  <- EAP-Request/
                                                  EAP-Type=EAP-TLS
                                              (TLS change_cipher_spec,
                                                   TLS finished)
      EAP-Response/
      EAP-Type=EAP-TLS ->
                              <- EAP-Success    <-Radius Access-Accept
```

Figure 3: Example message sequence

4. Manufacturer Dependancy and End-of-life

There is a valid concern about the over-reliance on the IoT device manufacturer for the initial network bootstrapping. After all, not all device manufacturers maybe willing or capable to run such an online service throughout the lifetime of the IoT device.

For example, the Revolv smart home hub lost all manufacturer support after the it was acquired [revolv]. As noted by [I-D.irtf-t2trg-iot-seccons], such end-of-life of devices may be planned or unplanned (for example when the manufacturer goes bankrupt or when the vendor just decides to abandon a product). A user should still be able to use/bootstrap/re-bootstrap this device. This can require some form of authorization handover.

Another question is whether there would be someone willing to continue offering an online AAA service for devices which are no longer supported by the original manufacturer. Whether this can be mandated by regulation or by having sufficient business incentives cannot be addressed in this draft. However, we would note that there are examples of open-source communities supporting existing devices irrespective of any manufacturer support. OpenWrt for home routers and Cyanogenmod (continued as Lineage OS) for smartphones are two such popular examples.

With some support from the IoT device manufacturer, the deployment models described thus far in this document are thankfully flexible enough to allow the user to choose an online AAA server different from the original device manufacturer. The RADIUS peering can simply be updated to reflect this change. For example, once the credentials for all the devices at aaa.examplevendor.com have been transferred to a new AAA server run by an open source community at aaa.openvendor.com, the RADIUS peering in can simply be updated to forward EAP requests from devices with NAI realm as examplevendor.com to aaa.openvendor.com at port 1812.

5. Alternative Manufacture Independent Deployment Models

In this section we look at some alternative deployment modes which don't rely on any pre-provisioned credentials of any sort on the IoT device. Consider the deployment architecture shown in Figure 4. While it looks similar to the figures above, the key difference is that the IoT device portal can now be any third-party online service rather than relying on manufacturer. As above, the AP is expected to forward all EAP authentication requests from IoT devices to a single online RADIUS server by setting up the necessary peering.

The user can now pre-register the credentials for new devices that will join the WiFi network. For example, the user can specify a secret that will be used by the device for joining the network. The device can then run EAP-PSK [RFC4764] with the online RADIUS server for securely joining the network. The online RADIUS server can prevent the user from registering the same (or similar) secrets for the different devices in the network. This would ensure that devices in network do not share the same secret.

```
                                             +-------------------+
                                             |        IoT        |
                        +-------------+      |   device portal   |
                        |Access Point|       | +-----------+     |
   +------------+       | +---------+ | RADIUS | |  RADIUS  |    |
   |            |       | | RADIUS +---------------+  Server  |  |
   |    IoT     | EAP   | | Client | |         | +-----+-----+   |
   |   Device   +-------+ +---------+ |        |       |         |
   |            |       |             |        | +-----+-----+   |
   +------------+       |             |        | |    AAA    |  |
                        +-------------+        | |   Server |   |
                                               | +-----------+   |
                                               +-------------------+
```

Figure 4: Deployment Architecture (Home)

Other EAP methods, such as EAP-NOOB [I-D.aura-eap-noob] can also be deployed with the architecture shown above. EAP-NOOB does not require any pre-provisioned credentials on the IoT device. Instead of requiring the user to input the same PSK on the two ends, the user simply transfers an out-of-band (OOB) message between the device and the server. This can be especially useful for IoT devices which lack the necessary user interface for entering PSKs.

6. Security Considerations

Fake device: It may seem that any device can simply join the users network because the attacker can always setup a fake registration portal and pretend to successfully authenticate every device. However, this is not really the case. Any device can be connected to the user's access point only if there is radius peering to the attackers registration portal.

There still remains the question of how does the device know which AP to try and connect to. To aid this discovery, it is necessary that IoT devices only use EAP methods that provide mutual authentication (such as EAP-TLS, EAP-PSK and EAP-NOOB).

The devices can then opportunistically try to connect with APs that are withing range (ignoring all open APs and APs that use WPA2-PSK). The devices would successfully connect to an AP, if it can forward the EAP messages to the right RADIUS server in the device portal. However, there may be scenarios where many APs setup peering with a few popular online IoT device portals. The devices in this case would connect to an unintended AP. While encryption of higher layer traffic is expected, this would still have negative consequences as IoT devices may inadvertently connect to and consume bandwidth from the wrong AP.

To prevent such inadvertent scenarios, additional information about the AP must be provided to the IoT device portal when the RADIUS peering is setup. The IoT device portal can then ask the user whether he/she wants to allow a new IoT device that is attempting to connect through his AP. Fortunately, such AP information can easily be communicated over RADIUS using, for example, the NAS-Identifier attribute.

7. IANA Considerations

There are no IANA impacts in this memo.

8. References

8.1. Normative references

[I-D.aura-eap-noob]
Aura, T. and M. Sethi, "Nimble out-of-band authentication for EAP (EAP-NOOB)", draft-aura-eap-noob-03 (work in progress), July 2018.

[I-D.irtf-t2trg-iot-seccons]
Garcia-Morchon, O., Kumar, S., and M. Sethi, "State-of-the-Art and Challenges for the Internet of Things Security", draft-irtf-t2trg-iot-seccons-15 (work in progress), May 2018.

[IEEE-802.1X]
Institute of Electrical and Electronics Engineers, "Local and Metropolitan Area Networks: Port-Based Network Access Control", IEEE Standard 802.1X-2004. , December 2004.

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

[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>.

[RFC4764] Bersani, F. and H. Tschofenig, "The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method", RFC 4764, DOI 10.17487/RFC4764, January 2007, <https://www.rfc-editor.org/info/rfc4764>.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <https://www.rfc-editor.org/info/rfc5216>.

[RFC6733] Fajardo, V., Ed., Arkko, J., Loughney, J., and G. Zorn, Ed., "Diameter Base Protocol", RFC 6733, DOI 10.17487/RFC6733, October 2012, <https://www.rfc-editor.org/info/rfc6733>.

[RFC6929] DeKok, A. and A. Lior, "Remote Authentication Dial In User Service (RADIUS) Protocol Extensions", RFC 6929, DOI 10.17487/RFC6929, April 2013, <https://www.rfc-editor.org/info/rfc6929>.

[RFC7542] DeKok, A., "The Network Access Identifier", RFC 7542, DOI 10.17487/RFC7542, May 2015, <https://www.rfc-editor.org/info/rfc7542>.

8.2. Informative references

[barbie] Gibbs, Samuel., "Hackers can hijack Wi-Fi Hello Barbie to spy on your children", November 2015, <https://www.theguardian.com/technology/2015/nov/26/hackers-can-hijack-wi-fi-hello-barbie-to-spy-on-your-children>.

[doorbell] Kumar, M., "How to Hack WiFi Password from Smart Doorbells", January 2016, <thehackernews.com/2016/01/doorbell-hacking-wifi-pasword.html>.

[nest] Nest, "Nest Learning Thermostat", January 2016, <https://nest.com/thermostat/meet-nest-thermostat/>.

[revolv] "Nest is permanently disabling the Revolv smart home hub", The Verge , April 2016, <https://www.theverge.com/2016/4/4/11362928/google-nest-revolv-shutdown-smart-home-products>.

[scale] Withings, "Body", January 2016, <https://www.withings.com/eu/en/products/body>.

[Sethi14] Sethi, M., Oat, E., Di Francesco, M., and T. Aura, "Secure Bootstrapping of Cloud-Managed Ubiquitous Displays", Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2014), pp. 739-750, Seattle, USA , September 2014, <http://dx.doi.org/10.1145/2632048.2632049>.

Author's Address

Mohit Sethi
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

EMail: mohit@piuha.net