IEEE P802.11
Wireless LANs

|  |
| --- |
| Opportunistic Wireless Encryption |
| Date: 2016-03-15 |
| Author(s): |
| Name | Affiliation | Address | Phone | email |
| Dan Harkins | Aruba Networks, an HP Company | 1322 Crossman avenue, Sunnyvale, California, United States of America | +1 408 225 4500 | dharkins at aruba networks dot com |
| Warren Kumari | Google, Inc. | 1600 Amphitheatre Parkway, Mountain View, California, United States of America | +1 650 253 0000 | warren at kumari dot net |

Abstract

This submission proposes an opportunistic encryption scheme for 802.11.

***Background***

*A BOF on enhancing the captive portal experience was held at IETF 93. Several problems with captive portals were identified, including using unencrypted 802.11 in conjunction with HTTPS to a captive portal, and the popular use of using a shared, public, and widely available PSK to enable encrypted 802.11. The latter was presented as having an additional issue in that it requires provisioning which can be problematic on the “customer”-side of what ends up being a free Wi-Fi service. There was a motivation to make it easier to deploy and use this free-but-still-encrypted network while not making the security any worse than a shared, public, and widely available PSK.*

*A proposal was made to advertise a new element in beacons and probe responses that indicate that the SSID is to be used as the PSK. While this definitely makes it easier to deploy, it arguably makes security worse because now the PSK is being advertised over the air and it is no longer necessary to do a dictionary attack in order to determine the keys used to protect 802.11 data frames. This submission describes an alternative idea: just do an unauthenticated Diffie-Hellman during association. This has the benefit of being easy to deploy (in fact, nothing is provisioned on either end) and offering better security because passive attack is no longer possible—the WPA-PSK attack tools will no longer work.*

*As a result of interest expressed at this BOF a new working group was formed at the beginning of 2016 in the Applications and Real-Time Area of the IETF entitled capport. The charter of capport covers secure protocols and mechanisms to interact with a captive portal. It is desirable though that 802.11 take up the work of OWE instead of passing it off to the capport WG. Hence this submission.*

*Opportunistic encryption schemes have been proposed for such security protocols as IKE and TLS/SSL. It is used for example, to protect SMTP traffic between mail relays to provide resistance to evesdropping of email when establishment of a trust relationship between relays proves difficult. Opportunistic encryption is described in RFC 7435. OWE is another kind of opportunistic encryption, this time for 802.11.*

*This is not a general purpose security mechanism but is proposed as a way to offer encrypted 802.11, in a manner that is more secure than using a shared and public PSK, when a separate authorization step—either clicking terms and conditions, watching a video, downloading an app, or whatever other step a user must do in order to obtain “free Wi-Fi”—is also available.*

*This is also not a proposal to replace, or be an alternative to, any existing mode of RSN security. It is instead aiming at a sweet spot that is underserved by the existing 802.11 security offerings—the desire to encrypt the air and do separate authorization. In this case, Open is unacceptable, PSK imposes provisioning requirements to configure a non-secret key that proves problematic in practice while also enabling attack that undermines the encryption, and 802.1X is too heavy-weight and imposes even more provisioning requirements on a user base that is not qualified to properly provision it. Due to the fact that OWE requires no provisioning, it could be thought of as an opportunistic encryption alternative to Open.*

***Instruct the editor to modify section 4.10.3.1 as indicated:***

**4.10.3.1 General**

This subclause summarizes the system setup and operation of an RSN in three cases— when a password or PSK is used during IEEE Std 802.11 authentication, when an IEEE Std 802.1X AS is used after Open System authentication, and when a PSK is used after Open System authentication—and the setup and operation of a non-RSN when a Diffie-Hellman exchange is opportunistically used during IEEE Std 802.11 (re)association. For an ESS, the AP includes an Authenticator, and each associated STA includes a Supplicant.

***Instruct the editor to create a new section 4.10.3.5:***

**4.10.3.5 Opportunistic Wireless Encryption**

The following AKM operations are carried out when opportunistic encryption is accomplished using a Diffie-Hellman key exchange.

* The STA discovers the AP’s security policy indicating support for Opportunistic Wireless Encryption through passively monitoring Beacon frames or through actively probing. After discovery in a non-DMG Infrastructure BSS, the STA performs Open System Authentication with the AP.
* The STA (re)associates with the AP and provides an ephemeral Diffie-Hellman public key in its (Re)Association Request frame. The AP responds with an ephemeral Diffie-Hellman public key in its (Re)Association Response frame.
* The STA and AP complete the Diffie-Hellman key exchange and generate a PMK.
* The 4-Way Handshake using EAPOL-Key frames is used, just as with IEEE Std 802.1X authentication, when an AS is present. See Figure 4-27 (Establishing pairwise and group keys).
* The GTK and GTK sequence number are sent from the AP to the STA just as in the AS case. See Figure 4-27 (Establishing pairwise and group keys) and Figure 4-28 (Delivery of subsequent group keys).
* If management frame protection is negotiated, the IGTK and IGTK packet number are sent from the AP to the STA just as in the AS case. See Figure 4-27 (Establishing pairwise and group keys) and Figure 4-28 (Delivery of subsequent group keys).

***Instruct the editor to modify sections 9.3.3.6-9.3.3.9 and replace ANA-1 with the next available number:***

**9.3.3.6 Association Request frame format**

 **Table 8-29—Association Request frame body**

|  |  |  |
| --- | --- | --- |
|  Order |  Information |  Notes |
| <ANA-1> |  Diffie-Hellman Parameter | The Diffie-Hellman Parameter is present if the OWE AKM is indicated in the RSNE in the Association Request frame. |

**9.3.3.7 Association Response frame format**

 **Table 8-30—Association Response frame body**

|  |  |  |
| --- | --- | --- |
|  Order |  Information |  Notes |
| <ANA-1> |  Diffie-Hellman Parameter | The Diffie-Hellman Parameter is optionally present if dot11RSNAActivated is true and the OWE AKM is advertised for the BSS. |

**9.3.3.8 Reassociation Request frame format**

 **Table 8-31—Reassociation Request frame body**

|  |  |  |
| --- | --- | --- |
|  Order |  Information |  Notes |
| <ANA-1> |  Diffie-Hellman Parameter | The Diffie-Hellman Parameter is present if the OWE AKM is indicated in the RSNE in the Reassociation Request frame. |

**9.3.3.9 Reassociation Response frame format**

 **Table 8-32—Reassociation Response frame body**

|  |  |  |
| --- | --- | --- |
|  Order |  Information |  Notes |
| <ANA-1> |  Diffie-Hellman Parameter | The Diffie-Hellman Parameter is optionally present if the OWE AKM is advertised for the BSS. |

***Instruct the editor to insert a new row in table 9-76 and request allocation of a number from ANA replacing “<ANA-2>” with the assigned number:***

**9.4.2.1 General**

 **Table 9-76—Element IDs**

|  |  |  |  |
| --- | --- | --- | --- |
|  Element |  Element ID |  Element ID Extension |  Extensible |
|  Diffie-Hellman Parameter |  255 |  <ANA-2> |  No |

***Instruct the editor to insert a new row in table 9-132 and request allocation of a number from ANA replacing “<ANA-3>” with the assigned number:***

**9.4.2.25.3 AKM Suites**

 **Table 9-132—AKM suite selectors**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  OUI |  Suite Type | Authentication Type |   Key Management type | Key derivation type |
|  00-0F-AC | <ANA-3> | Opportunistic Wireless Encryption  | RSNA key management defined in 12.7 (Keys and key distribution) or using PMKSA caching as defined in 12.6.10.3 (Cached PMKSAs and RSNA key management)  | Defined in 12.7.1.7.2 (Key derivation function (KDF)) using SHA-256 |

***Instruct the editor to create a new section 9.4.2.175:***

**9.4.2.175 Diffie-Hellman Parameter element**

The Diffie-Hellman Parameter element contains a Diffie-Hellman public value and an indicator of the finite cyclic group from which it was obtained. See figure 8-XYZ (Diffie-Hellman Parameter element).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  Element ID |  Length | Element ID Extension |  Group |  Public Key |

 Octets: 1 1 1 2 variable

The Element ID and Length fields are defined in 9.4.2.1 (General).

The Group field is a 16-bit unsigned integer that maps an identifying number from the “Group Description” registry maintained by IANA for IETF RFC 2409 (IKE) to a complete domain parameter set.

The Public Key is a Diffie-Hellman public key, an element in the group described by the domain parameter set indicated by the value in the Group field.

***Instruct the editor to modify section 12.6.10.3 as indicated:***

**12.6.10.3 Cached PMKSAs and RSNA key management**

In a non-FT environment, a STA might retain PMKSAs it establishes as a result of previous authentication. The PMKSA cannot be changed while cached. The PMK in the PMKSA is used with the 4-Way Handshake to establish fresh PTKs.

If a STA in an ESS has determined it has a valid PMKSA with an AP to which it is about to (re)associate, it performs Open System authentication to the AP, and then it includes the PMKID for the PMKSA in the RSNE in the (Re)Association Request. A non-AP STA may attempt PMKSA caching in conjunction with the OWE Authentication method by indicating OWE Authentication along with a PMKID in the RSNE and including a Diffie-Hellman parameter in the (Re)Association Request. If the AP has a PMKSA with a matching PMKID it shall not include a Diffie-Hellman parameter in its (Re)Association Response and not perform a Diffie-Hellman key exchange. If the AP does not have a PMKSA with a matching PMKID it shall perform OWE Authentication as if the PMKID was not present. When the PMKSA was not created using pre-authentication, the AKM indicated in the RSNE by the STA in the (Re)Association Request shall be identical to the AKM used to establish the cached PMKSA in the first place.

***Instruct the editor to create a new section 12.11, and the following sub-sections:***

**12.11 Opportunistic Wireless Encryption**

**12.11.1 General**

Opportunistic Wireless Encryption is executed between a non-AP STA and an AP to establish a PMKSA using a simple Diffie-Hellman key exchange. The exchange does not provide true authentication of the non-AP STA or AP but does allow for encryption. It is designed for cases in which access control is either not necessary or can be handled outside of this standard.

**12.11.2 Opportunistic Wireless Encryption exchange**

The Opportunistic Wireless Encryption (OWE) exchange occurs during association of a non-AP STA and an AP that indicate the OWE AKM (9.4.2.25.3 AKM Suites) by placing a Diffie-Hellman Parameter element in the (Re)Association Request and Response frames, respectively.

First the non-AP STA chooses a group in which to perform the Diffie-Hellman exchange. It then generates a secret private key, *m*, by randomly choosing it such that 1 < *m* < *r*, where *r* is the (prime) order of the chosen group. It derives a public key, ***M***, from the private key and the generator from the chosen group, ***G***:

 ***M*** = scalar-op(*m*, ***G***)

Where scalar-op() is defined in 12.4.4 (Finite cyclic groups). The non-AP STA then constructs a Diffie-Hellman Parameter element by assigning the appropriate value from the IANA-maintained registry for the chosen group to the Group field and assigning the public key to the Public key field using the conversion defined in 12.4.7.2.4 (Element to octet string conversion). The non-AP STA inserts the Diffie-Hellman Parameter element into its (Re)Association Request frame per 9.3.3.6 (Association Request frame format) or 9.3.3.8 (Reassociation Request frame format) and transmits the (Re)Association Request frame to the AP. If the non-AP STA chooses to do PMKSA caching it shall include a PMKID in the (Re)Association Request frame.

Upon receipt of the non-AP STA’s (Re)Association Request frame indicating OWE, the AP checks whether the STA is also attempting PMKSA caching. If so, and the AP has a PMKSA with a matching PMKID it shall proceed to the 4-way handshake as if it is not doing OWE, the STA’s Diffie-Hellman parameter is ignored. If it does not have a PMKSA with a matching PMKID it shall continue with OWE as if the PMKID was not present. The AP then checkes the Group field of the Diffie-Hellman Parameter element. If it is not an acceptable value the AP shall reject association with the reason of UNSUPPORTED\_FINITE\_CYCLIC\_GROUP. Otherwise it shall extract the non-AP STA’s public key from the Public Key field using the conversion defined in 12.4.7.2.5 (Octet string to element conversion). The extracted public key, ***M***, shall then be validated using the element validation technique from 12.4.5.4 (Processing of a peer’s SAE Commit message). If validation of the public key fails, the AP shall reject association with the reason of REQUEST\_DECLINED. If validation of the public key is successful, (Re)Association continues.

The AP shall then generate a private key, *n*, by randomly choosing it such that 1 < *n* < *r*, where *r* is the (prime) order of the group indicated in the Group field of the received Diffie-Hellman Parameter element. It derives a public key, ***N***, from the private key and the generator from the chosen group, ***G***:

 ***N*** = scalar-op(*n*, ***G***)

Where scalar-op() is defined in 12.4.4 (Finite cyclic groups). It shall then complete the Diffie-Hellman key exchange and generate a shared secret a PMK, and a PMKID:

 ***S*** = scalar-op(*n*, ***M***)

 *s* = F(***S***)

 *PMK* = KDF-Hash-Length(s, “OWE PMK Generation”, F(***M***) || F(***N***))

 *PMKID* = Truncate-128(SHA-256(F(***M***) || F(***N***)))

Where KDF-Hash-Length is the key derivation function defined in 12.7.1.7.2 (Key derivation function (KDF)) using the hash algorithm defined by the AKM in Table 9-132 to generate a key whose length is equal to the length of the hash algorithm’s digest, and F() is the element-to-scalar mapping function from 12.4.4 (Finite Cyclic groups). Intermediate data, ***S*** and *s*, shall be irretrievably deleted upon generation of the PMK. Upon generation of the PMKID the public keys ***M*** and ***N*** can be deleted. A PMKSA shall be created containing the PMK and PMKID.

The AP shall then construct an (Re)Association Response frame per 9.3.3.7 (Association Response frame format) or 9.3.3.9 (Reassociation Response frame format) indicating OWE. The AP constructs a Diffie-Hellman Parameter element with the Group field being identical to the Group field of the received (Re)Association Request frame and assigning its public key to the Public key field using the conversion defined in 12.4.7.2.4 (Element to octet string conversion). The AP inserts the Diffie-Hellman Parameter element into its (Re)Association Response frame per 9.3.3.7 (Association Response frame format) or 9.3.3.8 (Reassociation Response frame format) and transmits the (Re)Association Response frame to the STA.

Upon receipt of the AP’s (Re)Association Response frame indicating OWE, the STA checks that the Group field of the Diffie-Hellman Parameter element is identical to the value it inserted in its (Re)Association Request frame. If it is not the STA shall fail (Re)Association. Otherwise it shall extract the AP’s public key from the Public Key field using the conversion defined in 12.4.7.2.5 (Octet string to element conversion). The extracted public key, ***N***, shall then be validated using the element validation technique from 12.4.5.4 (Processing of a peer’s SAE Commit message). If validation of the public key fails, the STA shall fail (Re)Association.

Otherwise, the STA shall then complete the Diffie-Hellman key exchange and generate a shared secret a PMK, and a PMKID:

 ***S*** = scalar-op(*m*, ***N***)

 *s* = F(***S***)

 *PMK* = KDF-Hash-Length(s, “OWE PMK Generation”, F(***M***) || F(***N***))

 *PMKID* = Truncate-128(SHA-256(F(***M***) || F(***N***)))

Where KDF-Hash-Length is the key derivation function defined in 12.7.1.7.2 (Key derivation function (KDF)) using the hash algorithm defined by the AKM in Table 9-132 to generate a key whose length is equal to the length of the hash algorithm’s digest, and F() is the element-to-scalar mapping function from 12.4.4 (Finite Cyclic groups). Intermediate data, ***S*** and *s*, shall be irretrievably deleted upon generation of the PMK. Upon generation of the PMKID the public keys ***M*** and ***N*** can be deleted. A PMKSA shall be created using the PMK and PMKID.

**References:**