IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| An Authenticated Enryption Scheme for FILS Authentication | | | | |
| Date: 2012-11-04 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Dan Harkins | Aruba Networks | 1322 Crossman ave, Sunnyvale, CA | +1 408 227 4500 | dharkins at arubanetworks dot com |
|  |  |  |  |  |

Abstract

This submission proposes a scheme to perform authenticated encryption of Association frames for the FILS Authentication protocol using NIST-approved cipher modes.

***Instruct the editor to modify sections 11.9a.2.5 and 11.9a.2.6 as indicated:***

**11.9a.2.5 Encrypt and Authenticate operation for FILS Association frames**

The Encrypt and Authenticate operation takes an encryption key, KEK, an authentication key, KMK, plaintext to encrypt and authenticate, and additional associated data (AAD) to authenticate. The KEK and KMK shall be 128-bits. It uses AES-128 in CTR mode for encryption and AES-128 in CMAC mode for authentication. It outputs an authenticating tag, T, and ciphertext, C. The Encrypt and Authenticate function calls AES-CTR() and a function Authdata() to authenticate plaintext and the AAD.

AES-CTR takes an encryption key, a 128-bit initial counter value, and plaintext to encrypt; it outputs ciphertext of the same length as the input plaintext. AES-CMAC takes an authentication key and a variable-length string; it outputs a message authentication code (MAC) which authenticates the string using the key.The Authdata() function uses AES-CMAC. It takes an authentication key, plaintext, and the AAD; it outputs an authenticating tag. Authdata() uses a doubling function, dbl(), which multiplies a string by 2 modulo a primitive polynomial (it is the same function used in CMAC subkey generation), and a specialized xor function, xor-end, that XORs a string, x, onto the right-most length-of-x bits of a larger string, y.

Algorithmically, dbl(), Authdata() and EncryptAndAuthenticate() are as follows:

dbl(S)

{

if (MSB(S) == 0) then

S 🡨 S << 1

Else

S 🡨 ((S << 1) xor (0120 || 1 || 04 || 13)

Fi

Return (S)

}

Authdata(*KMK, plaintext, AAD*)

{

S 🡨 AES-CMAC(*KMK*, 0128)

S 🡨 dbl(S) xor AES-CMAC(*KMK*, *AAD*)

If (len(*plaintext*) >= 128) then

T 🡨 AES-CMAC(*KMK*, S xor-end *plaintext*)

Else

n = 128 – len(*plaintext*)

T 🡨 dbl(S) xor (*plaintext* || 1 || 0n-1)

T 🡨 AES-CMAC(*KMK*, T)

Fi

Return (T)

}

EncryptAndAuthenticate(*KEK, KMK, plaintext, AAD*)

{

T 🡨 Authdata(*KMK, AAD, plaintext*)

IV 🡨 T & (164 || 0 || 131 || 0 || 131)

C 🡨 AES-CTR(*KCK*, IV, *plaintext*)

Return (T || C)

}

where MSB(x) returns the most-significant bit of a string, x << y shifts a string, x, y-bits to the left, len(x) returns the length, in bits, of a string, || indicates concatenation, and xy indicates a string of the bit x that is y-bits long (if y is zero then the string is zero length)

**11.9a.2.6 Decrypt and Verify operation for FILS Association frames**

The Decrypt and Verify operation takes a decryption key, KEK, a verification key, KMK, an authenticating tag, a ciphertext to decrypt and verify, and additional associated data (AAD) to verify. It uses AES-128 in CTR mode for decryption and AES-128 in CMAC mode for verification. It outputs plaintext when all data has been verified or it outputs a failure indicator, FAIL, if some portion of the data—either the decrypted ciphertext or the AAD—was not verified.

AES-CTR and AES-CMAC are used in the same way as in section 11.9a.2.5. In addition, the Decrypt and Verify operation uses Authdata() from 11.9a.2.5.

Algorithmically, DecryptAndVerify() is as follows:

DecryptAndVerify(*KEK, KMK, C, T, AAD*)

{

IV 🡨 *T* & (164 || 0 || 131 || 0 || 131)

P 🡨 AES-CTR(*KCK*, IV, *C*)

V 🡨 Authdata(*KMK, AAD,* P)

If (V == *T*) then

Return (P)

Else

Return (FAIL)

Fi

}

**References:**