IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Correcting Some Mistakes | | | | |
| Date: 2015-06-25 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Dan Harkins | Aruba Networks | 1322 Crossman avenue, Sunnyvale, California, United States of America | +1 408 227 4500 | dharkins at aruba networks dot com |
|  |  |  |  |  |

Abstract

A conditional statement in the 11-14/0640r1 was missed when that document was incorporated into the TGmc draft. As a result the new hunting-and-pecking loop for ECC groups would not be backwards compatible with IEEE Std 802.11-2012.

Thanks to Jouni Malinen for discovering this issue before it was too late.

In addition, there are some other small mistakes in the SAE definition that should be fixed: 1) the commit-scalar needs to be greater than 1; 2) an FFC public key has to be between 1 and (p-1); and, 3) it is possible for the legendre symbol to be zero and that would cause the quadratic residue and quadratic non-residue to not be what we want. While the patient is on the operating table let’s take care of this.

***Instruct the editor to modify section 11.3.5.2 as indicated:***

**11.3.5.2 PWE and secret generation**

After generation of the PWE , each STA shall generate a secret value, rand , and a temporary secret value,

mask , each of which shall be chosen randomly such that 1 < rand < r and 1 < mask < r and (rand + mask) modulo r is greater than one (1), where r is the (prime) order of the group. If their sum modulo r is not greater than one (1), they shall both be irretrievably deleted and new values shall be randomly generated.

***Instruct the editor to modify section 11.3.5.4 as indicated:***

**11.3.5.4 Processing of a peer’s SAE Commit message**

If the scalar value is greater than zero (0) and less than the order, r , of the negotiated group, scalar validation succeeds; otherwise, it fails. Element validation depends on the type of group. For FFC groups, the element shall be an integer greater than one (1) and less than the prime number minus one, (p -1) , and the scalar operation of the element and the order of the group, r , shall equal one (1) modulo the prime number p . If either of these conditions does not hold, element validation fails; otherwise, it succeeds.

***Instruct the editor to modify section 11.3.4.2.2 as indicated:***

**11.3.4.2.2 Generation of the password element with ECC groups**

Algorithmically this process is described as follows:

*found* = 0;

*counter* = 1

*Length* = len(*p*)

*base* = *password*

do {

*pwd -seed* = H(MAX(STA-A-MAC, STA-B-MAC) || MIN(STA-A-MAC, STA-B-MAC),

*base* || *counter*)

*pwd -value* = KDF-Length(*pwd -seed* , “SAE Hunting and Pecking”, *p*)

if (*pwd-value* < *p* )

then

if (*pwd-value*3 + a x *pwd-value* + b ) is a quadratic residue modulo *p*

then

if (*found* == 0)

then

*x* = *pwd-value*

*save* = *pwd-seed*

*found* = 1

*base* = new-random-number

fi

fi

fi

*counter* = *counter* + 1

} while ((*counter* <= k ) or (*found* =0))

*y* = sqrt(*x*3 + a*x* + b ) modulo *p*

if LSB(*save*) = LSB(*y*)

then

PWE = (*x* , *y*)

else

PWE = (*x* , *p – y*)

fi

This technique involves creation of a quadratic residue, qr , and quadratic non-residue, qnr , prior to

beginning of the hunting-and-pecking loop. These values can be chosen at random by checking their

legendre symbol:

do {

*qr* = random() modulo *p*

while ( LGR(*qr* |*p*) is not equal to 1)

do {

*qnr* = random() modulo *p*

while ( LGR(*qnr* |*p*) is not equal to -1)

**References:**