IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Username SAE | | | | |
| Date: 2014-10-31 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Dan Harkins | Aruba Networks | 1322 Crossman ave, Sunnyvale, CA, United States of America | +1 408 227 4500 | dharkins at aruba networks dot com |
|  |  |  |  |  |

Abstract

This document proposes additions to the standard to support transmission of a username to identify a password used by the SAE protocol.

***Instruct the editor to modify table 8-44 in section 8.3.3.11 as indicated:***

**8.3.3.11 Authentication frame format**

**Table 8-44—Presence of fields and elements in Authentication frames**

|  |  |  |  |
| --- | --- | --- | --- |
| USAE | 1 | Status | Challenge Text is present if Status is zero  Scalar is present if Status is zero.  Element is present if Status is zero.  Anti-Clogging Token is present if status is 76 or if  frame is in response to a previous rejection with  Status 76.  Finite Cyclic Group is present if Status is zero or 76. |
| USAE | 2 | Status | Send-Confirm is present. Confirm is present. |
| USAE | 2 | 82 | One or more Neighbor Report element(s) is present |

***Instruct the editor to modify section 8.4.1.1 as indicated:***

**8.4.1.1 Authentication Algorithm Number field**

Authentication Algorithm number = 3; simultaneous authentication of equals (SAE)

Authentication Algorithm number = <ANA-1>; username simultaneous authentication of equals (USAE)

Authentication Algorithm number = 65535; Vendor specific use

***Instruct the editor to modify section 8.4.2.8 as indicated:***

**8.4.2.8 Challenge Text element**

The length of the Information field is dependent upon the authentication algorithm and the transaction

sequence number as specified in 11.2.3.2 (Open System authentication) and 11.3 (Authentication using a password).

***Instruct the editor to modify table 8-140 in section 8.4.2.24.3 as indicated:***

**8.4.2.24.3 AKM suites**

**Table 8-140—AKM suite selectors**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 00-0F-AC | <ANA-2> | Username variant of SAE authentication with SHA-256 or using PMKSA caching as defined in 11.5.10.3 (Cached PMKSAs and RSNA key management) with SHA-256 key  derivation | RSNA key management as defined in 11.6 (Keys and key distribution), PMKSA caching as defined in 11.5.10.3 (Cached PMKSAs and RSNA key management) with SHA-256 key derivation or authenticated mesh peering exchange as defined in 13.5 (Authenticated mesh peering exchange (AMPE)) | Defined in  11.6.1.7.2 (Key  derivation  function (KDF)) |
| 00-0F-AC | <ANA-3> | FT authentication over username variant of SAE authentication with SHA-256 | FT key management defined  in 11.6.1.7 (FT key hierarchy) | Defined in  11.6.1.7.2 (Key  derivation  function (KDF)) |
| 00-0F-AC | <ANA-3>+1-255 | Reserved | Reserved | Reserved |

***Instruct the editor to modify section 10.3.4.2 as indicated:***

**10.3.4.2 Authentication—originating STA**

Upon receipt of an MLME-AUTHENTICATE.request primitive, the originating STA shall authenticate

with the indicated STA using the following procedure:

b) The STA shall execute one of the following:

3) For SAE authentication, including the Username variant of SAE, in an ESS, IBSS, or MBSS, the authentication mechanism described in 11.3 (Authentication using a password).

***Instruct the editor to modify section 10.3.4.3 as indicated:***

**10.3.4.3 Authentication—destination STA**

c) If SAE authentication, including the Username variant of SAE, is being used in an ESS, IBSS, or MBSS, the MLME shall issue an MLMEAUTHENTICATE.indication primitive to inform the SME of the authentication request, including the SAE authentication elements, and the SME shall execute the procedure as described in 11.3 (Authentication using a password)

***Instruct the editor to modify section 11.1.2 as indicated:***

**11.1.2 Security methods**

RSNA security comprises the following algorithms and procedures:

* RSNA establishment and termination procedures, including use of IEEE Std 802.1X authentication, described in 11.5 (RSNA security association management) and SAE authentication, including the Username variant of SAE, described in 11.3 (Authentication using a password)

***Instruct the editor to modify section 11.1.4 as indicated:***

**11.1.4 RSNA establishment**

An SME establishes an RSNA in one of six ways:

b) If an RSNA is based on a PSK or password in an ESS, the SME establishes an RSNA as follows:

2) If the RSNA-capable AP advertises support for SAE authentication, including the Username variant of SAE, in its Beacon or Probe Response frames, and the STA has a group defined in the dot11RSNAConfigDLCGroupTable and a password for the AP, identified by the MAC address for SAE and the SSID for the Username variant of SAE, in the dot11RSNAConfigPasswordValueTable, the STA shall invoke SAE authentication to establish a PMK. If the RSNA-capable AP does not advertise support for SAE authentication in its Beacon and Probe Response frames but advertises support for the alternate form of PSK authentication (see 4.10.3.4 (Alternate operations with PSK)), and the STA also supports the alternate form of PSK authentication, the non-DMG STA may invoke Open System authentication and use the PSK as the PMK with the key management algorithm in step 4) below.

***Instruct the editor to modify section 11.3.1 as indicated:***

**11.3.1 SAE overview**

There are two variants of SAE defined, a peer-to-peer variant—indicated by the AKM of 8 or 9—and a client-server variant—indicated by the AKM of <ANA-2>. Both variants are described in this section as “SAE”. In the peer-to-peer variantSAE does not have a notion of an “Initiator” and “Responder” or of a “Supplicant” and “Authenticator.” The parties to the exchange are equals, with each side being able to initiate the protocol. Each side may initiate the protocol simultaneously such that each side views itself as the “initiator” for a particular run of the protocol. This requirement is necessary to address the unique nature of MBSSs. In the client-server variant a non-AP STA is viewed as the “Initiator” and the AP is viewed as the “Responder”.

The parties involved are called STA-A and STA-B. In the peer-to-peer variant, they are identified by their MAC addresses, STA-A-MAC and STA-B-MAC, respectively. In the client-server variant, they are identified by the usernames they convey to each other. STAs begin the protocol when they discover a peer, or AP, through Beacons and Probe Responses, or when they receive an Authentication frame indicating SAE or USAE authentication from a peer.

SAE is an RSNA authentication protocol and is selected according to 11.5.2 (RSNA selection).

The peer-to-peer variant of SAE shall be implemented on all mesh STAs to facilitate and promote interoperability.

***Instruct the editor to modify section 11.3.5.3 as indicated:***

**11.3.5.3 Construction of an SAE Commit message**

An SAE Commit message consists of a scalar and an element that shall be produced using the PWE and secrets generated in 11.3.5.2 (PWE and secret generation), as follows:

*commit -scalar* = (*rand* + *mask* ) modulo r

***COMMIT-ELEMENT*** = inverse(scalar-op(*mask* , ***PWE*** ))

When the AKM is <ANA-2>, a username is added to the Authentication frame in the Challenge Text field as described in 8.3.3.11 (Authentication frame format). The non-AP STA shall use its provisioned username and the AP shall use a provisioned username, if one exists, and the SSID associated with the BSSID to which the STA is authenticating if one does not exist. The username shall be encoded as a non-terminated ASCII string whose length shall be less than 254.

This message shall be transmitted to the peer as described in 11.3.7 (Framing of SAE). The temporary secret mask may be destroyed at this point.

***Instruct the editor to modify section 11.3.5 4 as indicated:***

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

Upon receipt of a peer’s SAE Commit message, the first step is to check against a reflection attack. If a Commit message has been sent, peer’s scalar and element shall be verified to be different than commit-scalar and COMMIT-ELEMENT. In addition, if the AKM is <ANA-2>, the contents of the Challenge Text element of the Authentication frame shall be verified to be different than the provisioned username (and SSID, for the AP). If either of these checks fails, SAE authentication shall fail.

Next, both the scalar and element shall be verified.

The entropy of k shall then be extracted using H to produce keyseed . The key derivation function from 11.6.1.7.2 (Key derivation function (KDF)) shall then be used to derive a key confirmation key, KCK, and a pairwise master key, PMK, from keyseed . When used with AKMs 8 or 9, the salt shall consist of thirty-two (32) octets of the value zero (0) (indicated below as <0>32) and both the KCK and PMK shall be 256-bits in length.

*keyseed* = H(<0>32, *k* )

(*KCK* || *PMK* ) = KDF-512(*keyseed* , “SAE KCK and PMK”,

(*commit-scalar* + *peer-commit-scalar* ) modulo *r* )

When used with AKM <ANA-2>, the salt shall consist of the addition of the two parties’ commit-scalar values modulo r, and both the KCK and PMK shall be 256-bits in length.

*keyseed* = H((*commit-scalar* + *peer-commit-scalar*) modulo *r*, *k*)

(*KCK* || *PMK*) = KDF-512(*keyseed*, “SAE KCK and PMK”,

*username-sta* || *username-ap*)

where username-sta is the contents of the Challenge Text element of the Authentication frame containing the non-AP STA’s Commit message and username-ap is the contents of the Challenge Text element of the Authentication frame containing the AP’s Commit message.

Use of other AKMs require definition of the lengths of the salt, the KCK, and the PMK.

The PMK identifier is defined as follows:

PMKID = L((*commit-scalar* + *peer-commit-scalar* ) modulo *r* , 0, 128)

***Instruct the editor to modify section 11.3.5.5 as indicated:***

**11.3.5.5 Construction of an SAE Confirm message**

A peer generates an SAE Confirm message by passing the KCK, the current value of the *send-confirm* counter (see 8.4.1.37 (Send-Confirm field)), the scalar and element from the sent SAE Commit message, and the scalar and element from the received SAE Commit message to the confirmation function CN. When the AKM used is <ANA-2>, the username of the transmitter of the SAE Confirm message is also passed to the function CN.

*confirm* = CN(**KCK**, *send-confirm*, *commit -scalar*, ***COMMIT-ELEMENT*** , *peer-commit-scalar*,

***PEER-COMMIT-ELEMENT [,*** *username****]***)

Where *username* is optionally appended when the AKM is <ANA-2>. The *send-confirm* counter shall be in the format specified in section 8.2.2 (Conventions) in the order in which it is transmitted over the air. The *username* shall be encoded as a string of octets in the order sent, if applicable, The message shall be transmitted to the peer as described in 11.3.7 (Framing of SAE).

***Instruct the editor to modify section 11.3.5.6 as indicated:***

**11.3.5.6 Processing of a peer’s SAE Confirm message**

Upon receipt of a peer’s SAE Confirm message a *verifier* is computed, which is the expected value of the peer’s confirmation, *peer-confirm*, extracted from the received an SAE Confirm message. The *verifier* is computed by passing the KCK, the peer’s send-confirm counter from the received an SAE Confirm message (see 8.4.1.37 (Send-Confirm field)), the scalar and element from the received SAE Commit message, and scalar and element from the sent SAE Commit message to the confirmation function CN. When the AKM used is <ANA-2>, the username of the transmitter of the SAE Confirm message is also passed to the function CN.

*verifier* = CN(**KCK**, *peer-send-confirm*, *peer-commit-scalar*, ***PEER-COMMIT-ELEMENT***,

*commit-scalar*, ***COMMIT-ELEMENT*** *[, username]*)

Where *username* is optionally appended when the AKM is <ANA-2>. The *peer-send-confirm* shall be in the format in section 8.2.2 (Conventions), as extracted out of the receive frame. The *username* shall be encoded as a string of octets in the order received, if applicable. If the *verifier* equals *peer-confirm*, the STA shall accept the peer’s authentication and set the lifetime of the PMK to the value dot11RSNAConfigPMKLifetime. If the *verifier* differs from the *peer-confirm*, the STA shall reject the peer’s authentication and destroy the PMK.

***Instruct the editor to modify section C.3 as indicated:***

**C.3 MIB Detail**

-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-- \* dot11RSNAConfigPasswordValue TABLE

-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

dot11RSNAConfigPasswordValueTable OBJECT-TYPE

SYNTAX SEQUENCE OF Dot11RSNAConfigPasswordValueEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"When SAE authentication is the selected AKM suite,

this table is used to locate the binary representation

of a shared, secret, and potentially low-entropy word,

phrase, code, or key that will be used as the

authentication credential between a TA/RA pair.

This table is logically write-only. Reading this table

returns unsuccessful status or null or zero."

::= { dot11smt 25 }

dot11RSNAConfigPasswordValueEntry OBJECT-TYPE

SYNTAX Dot11RSNAConfigPasswordValueEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry (conceptual row) in the Password Value Table"

INDEX { dot11RSNAConfigPasswordValueIndex }

::= { dot11RSNAConfigPasswordValueTable 1 }

Dot11RSNAConfigPasswordValueEntry ::=

SEQUENCE {

dot11RSNAConfigPasswordValueIndex Unsigned32,

dot11RSNAConfigPasswordIdentity OCTET STRING,

dot11RSNAConfigPasswordAPSSID OCTET STRING,

dot11RSNAConfigPasswordCredential OCTET STRING,

dot11RSNAConfigPasswordPeerMac MacAddress }

dot11RSNAConfigPasswordValueIndex OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The auxiliary variable used to identify instances of the columnar

objects in the Password Value table."

::= { dot11RSNAConfigPasswordValueEntry 1 }

dot11RSNAConfigPasswordPeerMac OBJECT-TYPE

SYNTAX MacAddress

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This is a control variable.

It is written by an external management entity.

Changes take effect as soon as practical in the implementation.

This variable represents the MAC address of the peer

that is to be authenticated using the peer-to-peer variant of SAE.

A wildcard BSSID ispermitted when passwords are shared among peers."

::= { dot11RSNAConfigPasswordValueEntry 3 }

dot11RSNAConfigPasswordIdentity OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1..253))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

“This is a control variable.

It is written by an external management entity.

Changes take effect as soon as practical in the implementation.

This variable is an ASCII representation of an identity used

by a STA using the client server variant of the SAE protocol.

The non-AP STA uses this to identify itself and the AP uses

this to look up the STA’s username in the SAE protocol.

Any ASCII character string is valid.”

:= { dot11RSNAConfigPasswordValueEntry 4 }

dot11RSNAConfigPasswordAPSSID OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1..32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

“This is a control variable.

It is written by an external management entity.

Changes take effect as soon as practical in the implementation.

This variable is the desired SSID for a non-AP STA to connect

to using the accompanying username and credential. For an AP

this variable is unused.”

:= {dot11RSNAConfigPasswordValueEntry 5 }

**References:**