802.16 Security and Encryption

by James Wiebe
for Dr. Erfani
88-590-18 Network Security, W2005
April 14, 2005
University of Windsor

Abstract – The cryptographic techniques used in the IEEE 802.16-2001 [5] standard and the algorithms built to use them, are described and illustrated by a C++ dialog-based program for each of the Authorization and TEK FSMs in the SS.

Summary

802.16 cryptography uses a two-tiered keying approach, using public key encryption minimally, only to establish the first-tier shared secret, the Authorization Key (AK). Keys exchanged (Traffic Encryption Keys or TEKs) using symmetric encryption based on keys derived from the AK (Key Exchange Keys KEKs) are used to encrypt message traffic and authentication is initially based on embedded X.509 certificates and routinely done via keys derived from the AK. RSA is used to share the AK and triple-DES is used for symmetric encryption (when the KEKs and TEKs are used). SHA-1 is used for HMAC (Hashed Message Authentication Code) digests. All symmetric keys are created by the Base Station (BS); it is the responsibility of the SS (Subscriber Station) to request these keys and keep them refreshed, since they continually time out and expire. They are kept in pairs and the newer AKs and TEKs overlap the old by generally half, although this is configurable.
Authorization and TEK state machines (FSMs: Finite State Machines) that run in the SS are event and message driven; the Authorization FSM starts and stops TEKs whereas events and messages that can be traced back to the operation of the TEKs control the SS Authorization FSM only via the BS.

Introduction

The 802.16 standard is a large one, taking up 349 pages in total. It specifies the operation of the OSI Physical layer and a lower part of the Data Link layer comprising a wireless metropolitan area network (MAN) standard. It is intended to scale to larger areas than does the popular 802.11 wireless Local Area Network (LAN) standard, which has been implemented as “WiFi”. Its Service Specific Convergence Sublayer is designed to support ATM (Asynchronous Transfer Mode) cells as well as packets; therefore one of its applications could be to support IP (internet protocol). This has the potential of realizing the dream of ubiquitous portable high-speed WWW (World-Wide Web) access. An industry group is currently working on implementing 802.16 as “WiMax” [3].

Although the 802.16-2001 standard has been updated and “obsoleted” by the 802.16-2004 standard, the latter is not yet available for public access via the IEEE “Get 802” program [1]. These standards are made publicly available via this program six months after release and the new version was completed only in October 2004, making it not quite timely for this paper, which is due in April 2005.

The following diagram gives an overview of the scope of the 802.16 standard, taken from Figure 1 in the 802.16-2001 standard ([5], pg 2):

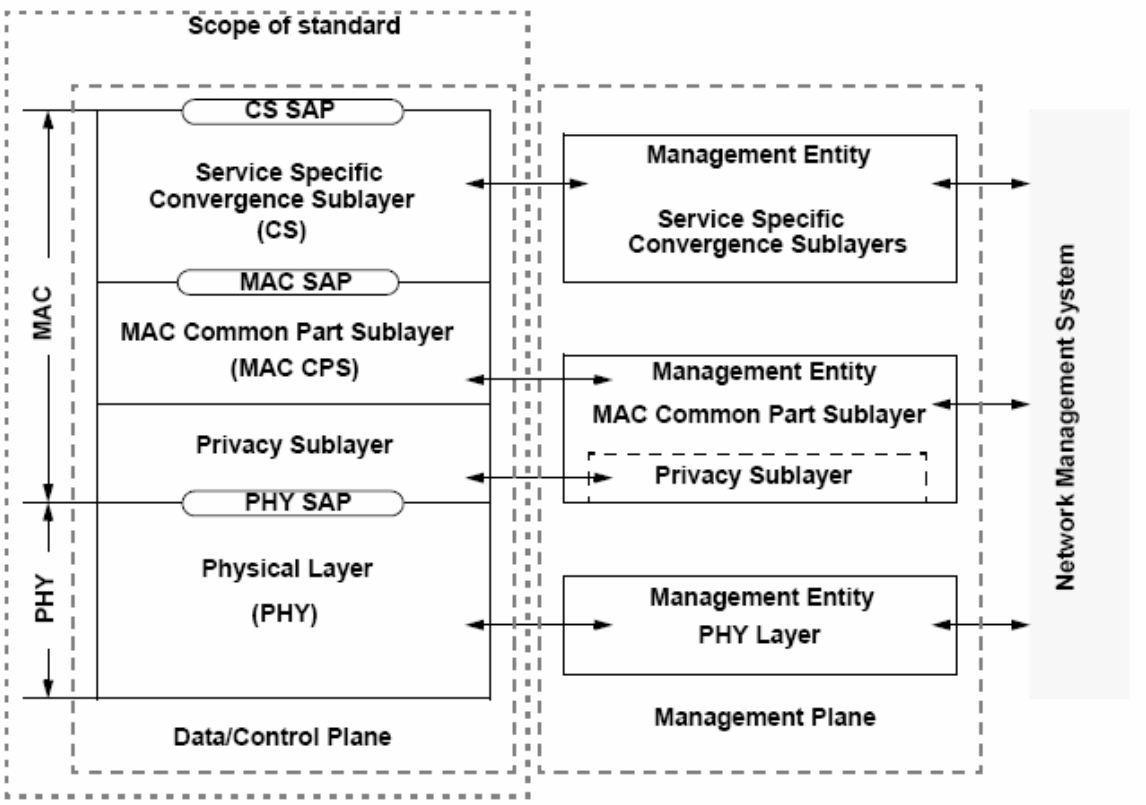

Figure 1. IEEE std. 802.16-2001 protocol layering.

"SAP" stands for "Service Access Point", each of which forms a conceptual "doorway" between an adjacent pair of sublayers. Each layer communicates with the same layer in the other entity and information passed down to it from the next higher layer is treated as "black-box" data that is only "looked at" at most in order to assure that it cannot interfere with the operation of the current layer. It can be amalgamated or split up into different packets before being passed down to the next lower layer. Data is received from above in the form of SDUs (Service Data Units) and processed by the current layer into PDUs (the Protocol Data Units of the current layer), which become in turn the SDUs of the next lower layer.

Wireless telephony, for one, has been notorious for its lack of privacy. Modern scrambling and then encryption techniques (scrambling algorithms can become public knowledge, thus their security usefulness is limited and their sole use is now somewhat old-fashioned) have since dealt with that problem, and most new communication standards require a privacy component. The reliable authentication made possible by these techniques also provides the vendor with the direct benefit of preventing theft of service.

This paper focuses on the operation of the Privacy sublayer. The following sections describe the Architecture, the Key Management Protocol, Key Usage, and Cryptographic methods.

1. Architecture

IEEE 802.16-2001 specifies an authenticated client/server key management protocol in which the BS acts as a server and controls distribution of keying material to the SS.

The privacy sublayer consists of two component protocols, a), a protocol for encrypting the packetized data across the connection and b), a key management protocol, called "PKM", for "Privacy Key Management", which defines the distribution of keying data from the BS to the SS. PKM uses a two-tier system in which the public key cryptography is reserved for the sharing of symmetric "Authorization Keys" (AKs), which in turn are used for the sharing of "Traffic Encryption Keys" (TEKs) which can be changed frequently using the AKs in 3-DES, in order to keep the computationally-intensive public-key cryptography to a minimum. Each BS and one or more SSs will share a "Security Association" (SA), each of which contains a "cryptographic suite" made up of TEKs and IVs (Initialization Vectors).

1.1 Packet Data Encryption

Encryption is done on the MAC PDU contents; not on the header; all MAC management messages shall be sent in the clear.

1.2 Key Management Protocol

X.509 digital certificates are used, along with RSA, and strong symmetric algorithms. The BS (Base Station) functions as a server and the SS (Subscriber Station) functions as a client. The two-tiered approach described in the introduction is used. The BS authenticates the SS during initial authorization; each SS carries an X.509 certificate which contains the SS's public key and MAC address. When requesting an AK (Authorization Key), the SS sends this certificate to the BS, which checks it and if the public key is successfully verified, uses it to encrypt the AK to send to the SS. The SS is allowed under 802.16 to generate its own RSA key pair, in which case the standard requires the provision of a method for obtaining an X.509 certificate.

1.3 Security Associations

A Security Association is the set of security information a BS and one or more of its SSs share. There are Primary, Static and Dynamic SAs. Each SS must establish a Primary SA with its BS; Static SAs are preplanned in the BS and Dynamic SAs are created and destroyed as needed to support specific service flows. The Primary SA must be unique to each SS, but the Static and Dynamic SAs can be shared between SSs.

An SA is made up of a cryptographic suite which can contain, for example, TEKs (Transmission Encryption Keys) and IVs (Initialization Vectors). SAs are identified by SAIDs. The SS requests the SA's keying material from the BS. The BS provides the lifetime remaining to that keying material when it sends it to the SS. The SS must refresh

its keying material before the old material expires, otherwise it will have to perform network reentry and be reauthorized.

1.4 Dynamic SAs

A Dynamic Security Association can be created by a BS in response to the enabling or disabling of specific service flows. The SSs are informed of these SAs via the exchange of DSx messages. The BS can dynamically establish SAs by sending an SA add message to an SS. When the SS receives such a message, it starts a TEK state machine for each SA indicated in the message. The SS or the BS can also request that the new service flow be handled by an existing SA, by sending the SAID of the desired SA in a DSA-REQ message to the other. In the case that this is initiated by the SS, the BS checks the SS's authorization and informs the SS. In the opposite case, the SS simply accepts the BS's mapping of the service flow onto the existing SA.

2. PKM (Privacy Key Management) Protocol

2.1 SS authorization and AK exchange overview

There are three steps to SS authorization, which is performed by the Authorization state machine:
-the BS authenticates the SS
-the BS provides the SS with an AK from which Key Encryption Keys (KEK) and message authentication keys are derived.
-the BS provides the SS with the SAIDs of primary and static security associations.

After achieving initial authorization, the SS periodically seeks reauthorization from the BS; this process is handled by the Authorization FSM. The SS must maintain its authorization in order to be able to update its TEKs so that it can communicate. TEKs are updated via the TEK FSM.

The authorization process begins with the SS sending an Authentication information message to it BS, which contains the SS's X.509 certificate. This message is strictly informative and the BS can choose to ignore it. Next, the SS sends an Authorization Request message by which it asks for its AK and any SAIDs of any static IDs that it should have from the BS. This message also contains its X.509 certificate, a list of cryptographic suite IDs for cryptographic methods the SS supports, and the SS's basic CID (Connection Identifier), which is equal to the primary SAID.

The BS then checks the certificate; if it verifies, the BS determines the cryptographic suite it shares with the SS, creates an AK, encrypts it with the SS's public key, and sends the AK in an Authorization Reply message. The Auth. Reply also includes a four-bit sequence number, so that successive AKs are numbered mod. 16, the key lifetime and the SAIDs for which the SS is authorized to obtain keying information.

The SS and BS can support two simultaneously active AKs, with overlapping lifetimes, so that the SS can refresh its TEK keying information without interruption.

2.2 TEK exchange overview

When an SS is authorized, it starts a TEK (Transmission Encryption Key) state machine for each of the SAIDs identified in the Authorization Reply message. These TEK state machines continually send Key Request messages to the BS, requesting refresh of their TEKs. The BS replies with a Key Reply message, containing a TEK which is triple-DES encrypted via (EDE) using a Key Encryption Key (KEK) derived from the AK, as well as a CBC IV and the remaining lifetime of the keying material. The BS maintains two active sets of keying material per SAID. The lifetimes of the two sets of keys overlap so that each set becomes active halfway through the lifetime of its predecessor and expires halfway through the lifetime of its successor. The BS includes both sets of an SAID's key material in its key reply. The SS uses the lifetime information to time its key requests so that it doesn't run out of valid TEKs to use.

A TEK state machine remains active as long as the SS has a valid AK, and the SS is authorized to operate in that particular SA, in which case the BS continues to provide fresh keys every key refresh cycle. If the SS receives an Authorization Reject message from the BS, it stops all of its TEK state machines. Individual TEK state machines can be started or stopped under the control of an SS's static SAIDs.

Communication between Authorization and TEK state machines occur through the passing of events and messages. The Authorization state machine generates events, such as Stop, Authorized, Authorization Pending and Authorization Complete, for the TEK FSMs under its control. Communication in the other direction is done indirectly via the BS: the BS may send a message for the SS Authorization FSM as a result of a request it received that was generated by the SS TEK FSM.

2.3 Authorization State Machine

The Authorization state machine consists of six states and eight events, including receipt of messages, that can trigger state transitions (see Figures 2 and 3).

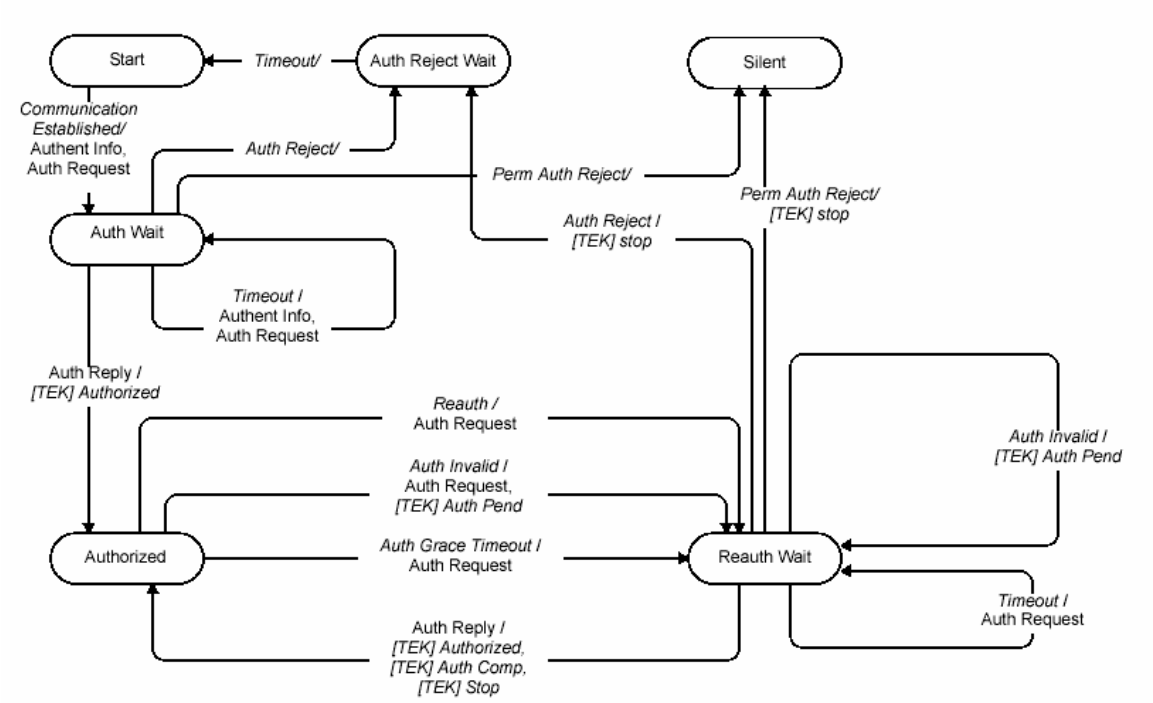

Figure 2. Authorization State Machine flow

In Figure 2, states are in ovals, events are in italics, and messages are in plain font. Lines are labeled in the format cause/response, where either can be events or messages. "[TEK]" indicates an event signaled to the TEK.

State ***Event or Rcvd Message***	**(A) Start**	**(B) Auth Wait**	**(C) Authorized**	**(D) Reauth Wait**	**(E) Auth Reject Wait**	**(F) Silent**
(1) *Communication Established*	Auth Wait					
(2) *Auth Reject*		Auth Reject Wait		Auth Reject Wait		
(3) *Perm Auth Reject*		Silent		Silent		
(4) Auth Reply		Authorized		Authorized		
(5) *Timeout*		Auth Wait		Reauth Wait	Start	
(6) *Auth Grace Timeout*			Reauth Wait			
(7) *Auth Invalid*			Reauth Wait	Reauth Wait		
(8) *Reauth*			Reauth Wait			

Figure 3. Authorization state machine transition matrix

In Figure 3, the allowed events for each state are shown in a white box that contains the new state to which the FSM will move as a result of the event.

2.4 TEK State Machine

The TEK consists of six states and nine events; see Figures 4 and 5.

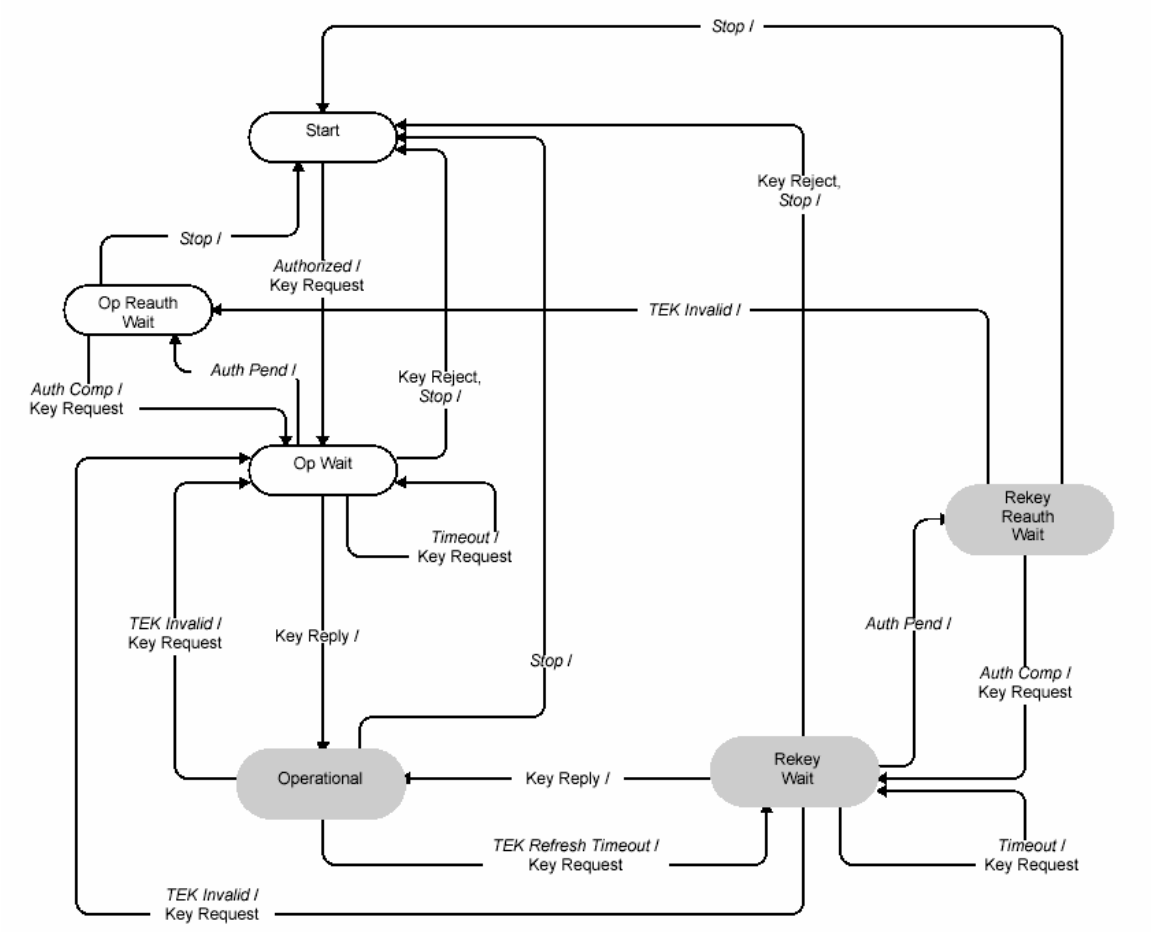

Figure 4. TEK FSM flow

State *Event or Rcvd Message*	(A) Start	(B) Op Wait	(C) Op Reauth Wait	(D) Op	(E) Rekey Wait	(F) Rekey Reauth Wait
(1) *Stop*		Start	Start	Start	Start	Start
(2) *Authorized*	Op Wait					
(3) *Auth Pend*		Op Reauth Wait			Rekey Reauth Wait	
(4) *Auth Comp*			Op Wait			Rekey Wait
(5) *TEK Invalid*				Op Wait	Op Wait	Op Reauth Wait
(6) *Timeout*		Op Wait			Rekey Wait	
(7) *TEK Refresh Timeout*				Rekey Wait		
(8) Key Reply		Operational			Operational	
(9) Key Reject		Start			Start	

Figure 5. TEK FSM state transition matrix

2.5 State Machine Illustration

Microsoft Visual C++ version 6.0 provides a convenient way to demonstrate the state machines, using its “dialog box” feature, which can be built via some clicks of the mouse [6]. The outlines of C++ classes [7] can be added to the code by the clicks of the mouse. I have implemented buttons on the dialog box, labeled with the events or messages that occur from the BS to the SS, to step the simulation through its paces, at the rate of one click per BS-SS communication. I have maintained the state of the SS state machine in a variable contained in the dialog box C++ class (i.e. in volatile memory) and its value is shown on-screen. A list box is added so that any state can be selected at will, which facilitates easy exploration of the state machine. The contents of the messages sent are kept in global variables to simulate over-the-air operation, and are shown on-screen. A log file showing the states and messages is generated, with buttons added for its display. The log file is closed and left on the computer HD when the program is exited; it is only erased if the user explicitly clicks on “Restart and Clear Log”.

A screen capture is shown (Figure 6, below) of the Authorization state machine C++ illustration. A similar C++ dialog-based illustration has been programmed for the TEK state machine.

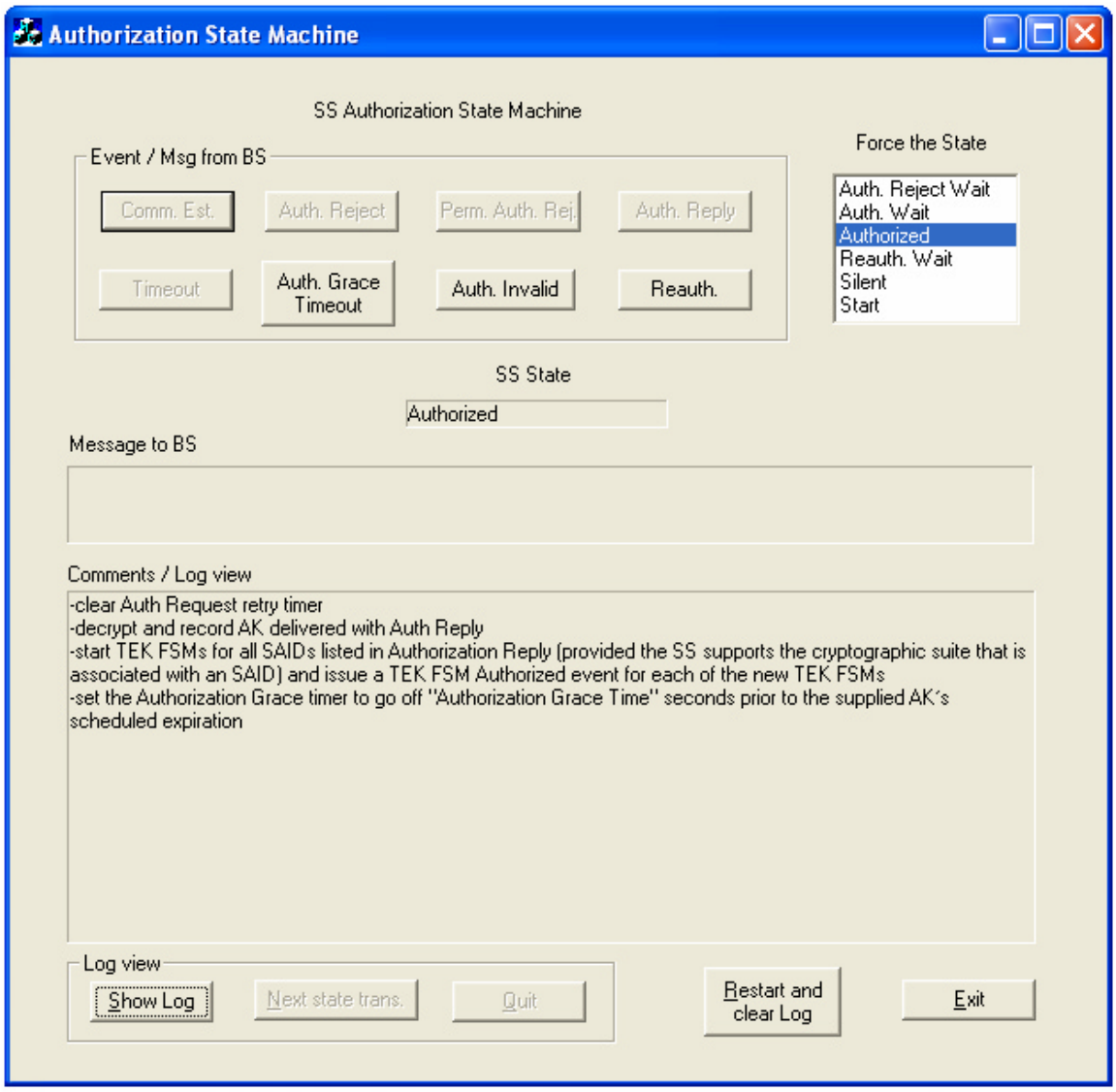

Figure 6. Authorization FSM C++ dialog-based illustration

3. Key Usage

The BS maintains keying information for all of the SAs.

After an SS completes basic capabilities negotiation, it initiates the authorization procedure with the BS. When the authorization reply is sent, the remaining lifetimes of the AKs are included, as accurately as possible. If an SS fails to reauthorize before the expiration of its current AK, the BS considers the SS unauthorized and removes from its keying tables all TEKs associated with the SS's Primary SA.

The BS will always send an AK to an authorized SS upon request. The overlapping AK key lifetimes are handled as follows (Figure 7): when the BS sends a second AK, it sets its lifetime equal to one "AK Lifetime" beyond the expiration of the older key. As long as

the BS is in the midst of an SS's AK transition period, it responds to the SS's Auth. Request messages with the newer of the two active keys. When the older key expires, an Auth Request will trigger the activation of a new AK and the start of a new key transition period. The updating of the TEK is handled in a similar manner.

3.1 Usage of AKs

The BS uses keying material derived from the AK that it assigns to the SS for three purposes: verifying the HMAC digests in key request messages received from the SS, creating the HMAC digests it writes into Key Reply, Key Reject and TEK Invalid messages sent to the SS, and encrypting the TEK in the Key Reply messages sent to the SS. HMAC_KEY_U is used to verify the messages received from the SS, and HMAC_KEY_D is used to create its HMAC digests ("U" stands for "Uplink" and "D" for "Downlink"). A KEK derived from the AK is used to encrypt the TEK in the Key Reply messages.

HMAC_KEY_D=SHA(H_PAD_D|AK)
HMAC_KEY_U=SHA(H_PAD_U|AK).
with
H_PAD_D=0x3A repeated 64 times
H_PAD_U=0x5C repeated 64 times.

KEK=Truncate(SHA-1(K_PAD_KEK | AK),128)
where K_PAD_KEK=0x53 repeated 64 times, i.e., a 512 bit string.
Truncate(x,*n*) denotes the result of truncating x to its left-most *n* bits.
SHA-1(*x*|*y*) denotes the result of applying the SHA-1 function to the concatenated bit strings *x* and *y*.

The key sequence number contained in the Key Request message allows the BS to determine the latest AK that the SS has successfully received; this is known as *implicit acknowledgement* by the SS that it has received that AK. The BS uses the latest AK that has been so acknowledged in its encryptions to the SS.

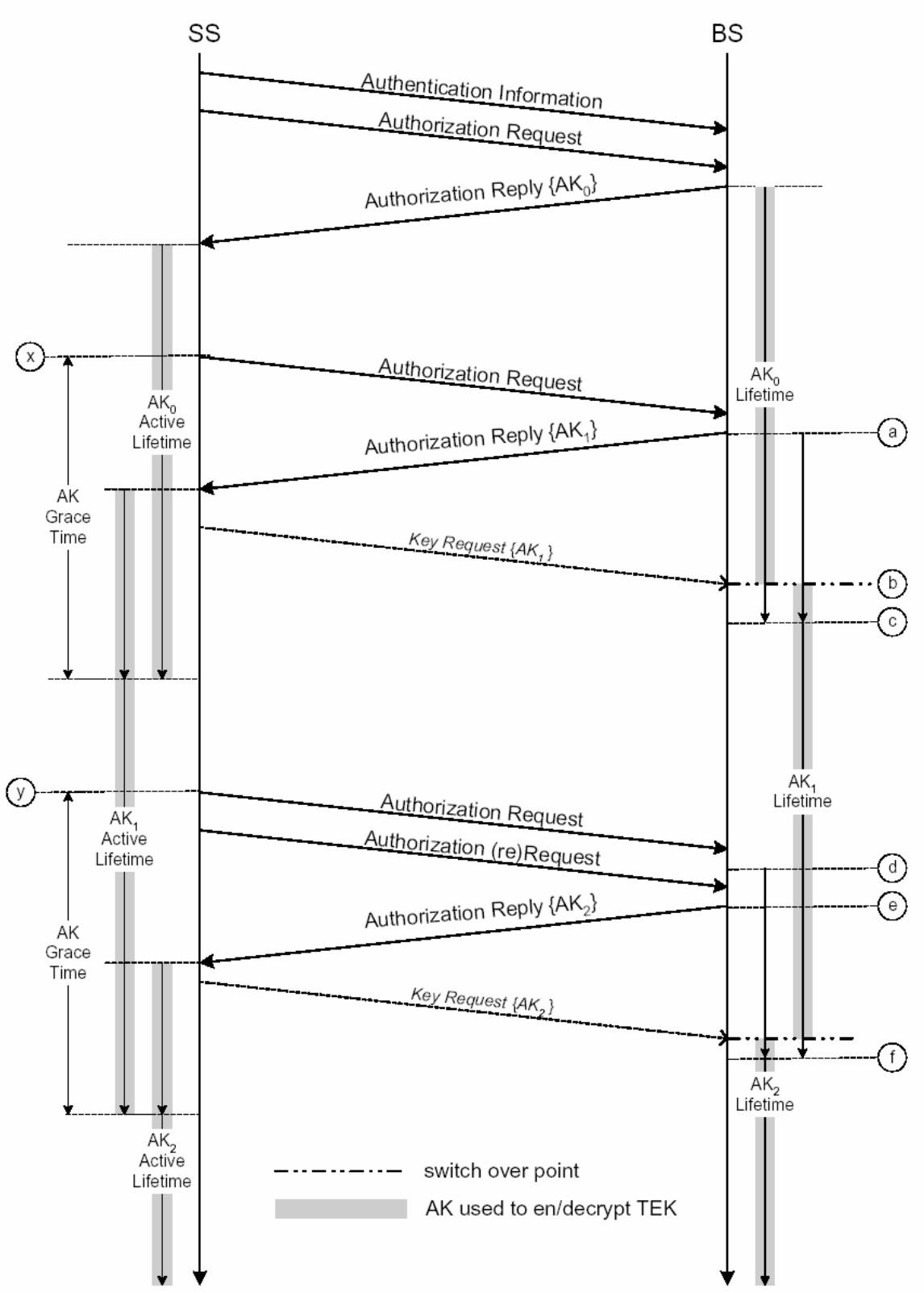

Figure 7. AK management

The SS uses HMAC_KEY_U calculated from its most recent AK when calculating the HMAC-Digests it attaches to Key Request messages, and it has the ability to use either AK to authenticate or decrypt received messages, determining the one to use from the Key Sequence number sent.

3.2 Usage of TEKs

For each of its SAs, the BS just simply switches over to using the new TEK when the previous one expires and it is the responsibility of the SS to make sure it has the new TEK. The transition period runs from the time that the BS sends the new TEK in a Key Reply message to the time that the older TEK expires. The BS uses the older of the two TEKs for encrypting downlink traffic and is able to use either to decrypt uplink traffic. An implication of this to note is that the BS only encrypts with a given TEK for the second half of that key's total lifetime.

The SS requests its new TEK a configurable amount of time, the *TEK Grace Time*, before the SS's latest TEK is set to expire. For each of its authorized SAIDs, the SS uses the newer of its two TEKs to encrypt uplink traffic and is able to use either of its TEKs to decrypt downlink traffic. This means that the SS only encrypts with a given TEK for the first half of that key's total lifetime.

4. Cryptographic Methods

The Data Encryption Algorithm identifier in the Cryptographic Suite of an SA normally will be set to 0x01 which will mean that data on connections associated with that SA shall use the Cipher Block Chaining mode (CBC) of DES (FIPS 46-3, 74, 81) to encrypt the MAC PDU payloads. The CBC IV will be calculated from the XOR of the IV parameter included in the TEK keying information and the content of the PHY Synchronization field of the latest DL-MAP. "Residual termination block processing" is used to encrypt the final block of plaintext when the final block is less than 64 bits – the XOR of the re-encrypt using ECB of the second last block with the plaintext of the last block; therefore the decryption of the final block is done by a second XOR of the ECB re-encrypt with the final block's ciphertext.

TEKs are encrypted by the BS to the SS using two-key triple-DES in EDE (Encrypt-Decrypt-Encrypt] mode:
encryption: $C = E_{k1}[D_{k2}[E_{k1}[P]]]$
decryption: $P = D_{k1}[E_{k2}[D_{k1}[C]]]$
P = Plaintext 64-bit TEK
C = Ciphertext 64-bit TEK
k1 = left-most 64 bits of the 128-bit KEK
k2 = right-most 64 bits of the 128-bit KEK
E[ ] = 56-bit DES ECB (electronic code book) mode encryption
D[ ] = 56-bit DES ECB decryption

The BS generates AKs, TEKs and IVs using a pseudo – random – number generator. Recommended practices for generating random numbers for use in cryptography can be found in RFC 1750.

5. Conclusions

Repeating the Summary (qv) would be tedious; we can say that the cryptographic protocols specified by 802.16 are fairly simple in concept, but can lead to considerable complexity in real-world applications, given multiple service flows.

References

Websites (sorted in order of base domain name)

[1]http://standards.ieee.org/getieee802/ - The website of the IEEE "Get 802" program. IEEE 802 standards are made available to the public, free of charge, six months after release.

[2] http://www.wikipedia.org – As an online free encyclopedia to which all are invited to contribute, it is rich in information on contemporary computer systems and computer-related mathematics, as well as many other areas of knowledge. It is a good place to look for explanations of acronyms the knowledge of which is taken for granted in electronic trade publications.

[3] http://www.wimaxforum.org – The website of the WiMax forum. "The WiMAX Forum is an industry-led, non-profit corporation formed to promote and certify compatibility and interoperability of broadband wireless products. Our member companies support the industry-wide acceptance of the IEEE 802.16 and ETSI HiperMAN wireless MAN standards."

[4] http://wirelessman.org – This is the website of the IEEE 802.16 working group on broadband wireless access standards.

Documents

[5] 802.16™ (2001) - IEEE Standard for Local and metropolitan area networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems, IEEE, http://www.ieee.org, New York, NY, 2002 – 349 pages.
This is the standard in all its glory, that implementers can flesh out.

Books

[6] Chapman, Davis, and Bates, John, Teach Yourself Visual C++ 6 in 21 Days, SAMS, a division of Macmillan Computer Publishing, http://www.mcp.com, Indianapolis, 1998, 0-672-31240-9
This book gives the follower the idea of how to get started programming with Windows in C++, i.e., the visual aspect, very quickly.

[7] Malik, D.S., C++ Programming - Program Design Including Data Structures, first ed., Course Technology, http://www.course.com, 2002, ISBN (2nd ed., March 23, 2004) 0-619-16044-6
This is an excellent book in explaining the detail of generic (non-visual) C++ programming to the beginner, in extremely minute detail. Also a good intermediate-level reference.