Modeling and Analysis of IEEE 802.16 PKM Protocols using CasperFDR

Sen Xu, Chin-Tser Huang, Manton M. Matthews

Department of Computer Science and Engineering
University of South Carolina
Columbia, SC 29208
Telephone: (803) 777–3285, Fax: (803) 777–3767
{xu4, huangct, matthews}@cse.sc.edu

***Abstract*—IEEE 802.16 is the standard for broadband wireless access. The security sublayer is provided within IEEE 802.16 MAC layer for privacy and access control, in which the Privacy and Key Management (PKM) protocols are specified. This paper models the PKM protocols using Casper and analyzes the CSP output with FDR, which are formal analysis tools based on the model checker. Later versions of PKM protocols are also modeled and analyzed. Attacks are found in each version and the results are discussed.**

I. Introduction

IEEE 802.16 is the standard for specifying the air interface of Wireless Metropolitan Area Network (WirelessMAN). The first version, IEEE 802.16-2001 [1], aimed to provide the last mile access for fixed Broadband Wireless Access (BWA), but assumed line-of-sight (LOS). IEEE 802.16-2004 [2], which supercedes several previous standards and amendments, also supports non-line-of-sight (NLOS) and mesh nodes. IEEE 802.16-2004 is sometimes also referred as WiMAX, which is in fact, a forum organized by industry to promote and certify products related to IEEE 802.16. IEEE 802.16e [3] adds mobility functionality to BWA, and is also known as mobile WiMAX.

The IEEE 802.16 standard specifies a security sublayer at the bottom of the MAC layer, to provide subscriber stations (SS) with privacy and to protect base stations (BS) from service hijacking. There are two component protocols in the security sublayer: an encapsulation protocol for encrypting packet data across BWA; a Privacy and Key Management protocol for secure distribution of keying information from the BS to the SS, and for enforcing authorized access to network services by BS.

In IEEE 802.16e, a new protocol PKM version 2 (PKMv2) is proposed, which adds an extensible authentication protocol (EAP) framework besides a revised RSA-based authentication. Since EAP only provides a framework, and the PKM protocol in IEEE 802.16-2004 (also referred as PKMv1) relates to RSA-based authentication, we analyze only the RSA-based authentication in PKMv2 in this paper.

Communicating Sequential Processes (CSP) is an algebra to describe the interactions in concurrent systems. CSP was first described by Hoare in [4] and [5], and has been applied in many fields. Roscoe applied it to modeling communication protocols [6], and proposed to verify the models with Failure Divergence Refinement (FDR) [7]. Schneider also modeled security properties in protocols with CSP [8], with more details about timed models.

Modeling and analysis of security protocols with CSP and FDR have been proven to be effective and have helped the research community find attacks in several protocols. However, modeling directly in CSP is time-consuming and error-prone. Lowe thus designed Casper [9], which takes more abstract descriptions of protocols as input and translates them into CSP. It is the aim of this paper to verify the security properties and discover the vulnerabilities of PKM protocols by modeling them with Casper and analyzing the CSP output with FDR.

The contribution of this work is fourfold. First, we formally model and analyze different versions of the PKM protocol with CasperFDR. Compared to BAN logic which requires error-prone manual modeling and has deficiency dealing with secrecy properties, CasperFDR is a more systematic and powerful tool. In particular, we show an attack which can be found with CasperFDR but not by BAN logic. Second, we verify that the attacks found with BAN logic in our previous work [10] are real threats as they are also found by CasperFDR. Third, we use CasperFDR to show that there are no other known attacks on PKM protocols. Since BAN logic and CasperFDR are two quite different formal methods, we believe our previous work and this work provide us confidence to claim that our proposed revision fixes the security problems of the PKM protocol and is not subject to other known attacks. Last but not least, we show that there are some problems with CasperFDR when used to analyze wireless authentication protocol. We discuss the obstacles we encountered and some feasible solutions to get around them. We hope this finding is helpful for the improvement of CasperFDR in the future.

The rest of the paper is organized as follows. In Section II, we introduce some related works. We model and analyze the original PKMv1, PKM Intel Nonce Version [11], PKMv2, and our Timestamp Version [12] with CasperFDR, in Section III, Section IV, Section V and Section VI respectively. Finally, we conclude in Section VII.

II. Related Works

Since the first version of the IEEE 802.16 standard [1] was released in 2002, many articles and books have been published. Johnson and Walker are among the first researchers that discuss the security issues in IEEE 802.16 [11]. They propose to enhance PKMv1 protocols with mutual authentication to enable the SS to authenticate the BS as well as the BS authenticating the SS, and with the addition of nonces to counter replay attacks. We refer to this work as the Intel Nonce version henceforth because the two authors are with Intel.

In our previous paper [12], we have analyzed security issues on the PKMv1 protocols and proposed solutions. We refer to our revised protocol as the Timestamp version because we suggest using timestamps to counter replay attacks. Following that, PKMv2 was proposed in the 802.16e standard, and we found a new attack on PKMv2 in [10]. Coincidently, another article [13] published the same attack shortly after our paper. There are also some dissertations dealing with PKMv2, such as [14]. We did not find much research on this protocol till now. However, there are many papers on protocols based on X.509, such as [15], [16], [17], and [18].

We also made formal analysis of PKMv1, PKMv2 and other versions using BAN logic in [10]. However, BAN logic has several deficiencies, such as the inability to handle secrecy properties, which is required by the key exchange protocols. Modeling and analysis with BAN logic is performed manually and is tedious and error-prone. Therefore, researchers have developed some automatic methods. CSP/FDR is one of them, and arguably one of the most successful so far. Casper was developed by Lowe [9], which takes abstract notations of protocols as input and translates them into CSP. FDR is adopted as the automatic tool to check the CSP model. Lowe has successfully discovered an attack using CSP and FDR on a protocol which had been regarded as safe for many years [19]. With Casper, Lowe and his students modeled and analyzed a library of protocols and discovered many attacks [20].

In this paper, we model PKM protocols in Casper and analyze their CSP output with FDR.

III. Modeling and Analyzing PKMv1

A. IEEE 802.16 PKMv1 Protocol

An SS begins authorization by sending an Authentication Information message which contains the SS manufacturer's X.509 certificate. This message is largely informative and the BS may choose to ignore it. Afterwards the SS sends an Authorization Request message (Auth-REQ) to its BS. In the response to the Auth-REQ, the BS validates the requesting SS's identity, determines the encryption algorithms and protocols to be shared with the SS, generates an Authentication Key (AK), and sends the AK to the SS. The authentication protocol is illustrated in Fig. 1.

B. Modeling PKMv1 in Casper

Now we model PKMv1 authentication protocol in Casper. In this model and later models, we use the conventional notations for agents(users). The initiator Alice (A) and the responder

- Message 1. $SS \rightarrow BS : Cert(SS.Manufacture)$
- Message 2. $SS \rightarrow BS :$ $Cert(SS) \mid Capabilities \mid BCID$
- Message 3. $BS \rightarrow SS :$ $KU_{SS}(AK) \mid SeqNo \mid Lifetime \mid SAIDList$

Fig. 1. Authentication Protocol in 802.16–2001

Bob (B) represent the SS and the BS respectively; Sam (S) represents the server; Kb and Km are the AK generated by Bob and Mallory (the Intruder) respectively. The PKMv1 model is shown as follows:

```
#Free variables
A, B : Agent
pka : PublicKey
PK : Agent -> PublicKey
SK : Agent -> SecretKey
kb, km : SessionKey
S : Server
pks : ServerPublicKey
sks : ServerSecretKey
InverseKeys = (PK,SK), (pks, sks), (kb, kb), (km, km)

#Processes
INITIATOR(A, S, pks) knows SK(A)
RESPONDER(B, S, kb, pks) knows SK(B)
SERVER(S, sks) knows PK

#Protocol description
0.    -> A : B
[A != B]
1.  A -> S : A
2.  S -> A:  {A, {A, PK(A)%pka}{sks}%certa}{sks}
3.  A -> B : certa % {A, PK(A) % pka}{sks}
[B !=A]
4.  B -> A : {kb}{pka % PK(A)}

#Specification
WeakAgreement(A, B)
WeakAgreement(B, A)
Secret(A, kb, [B])
Secret(B, kb, [A])

#Actual variables
Alice, Bob, Mallory : Agent
Kb, Km : SessionKey
PKs : ServerPublicKey
SKs : ServerSecretKey
Sam : Server
InverseKeys = (PKs, SKs), (Kb, Kb), (Km, Km)

#Inline functions
symbolic PK, SK

#System
INITIATOR(Alice, Sam, PKs)
RESPONDER(Bob, Sam, Kb, PKs)
SERVER(Sam, SKs)

#Intruder Information
Intruder = Mallory
IntruderKnowledge = {Alice, Bob, Mallory, Sam, \
Km, PKs, SK(Mallory)}
```

C. Analysis of PKMv1 with FDR

After compiling the above Casper model and feeding the CSP output to FDR, attacks were found to each of the four assertions declared in the Specification part. Using the debug mode in FDR, we could find the traces for those attacks. Following is the trace for the attack on asssertion Secret(A, kb, [B]):

```
<receive.Alice.Sam.(Msg1,Alice,<>)
send.Sam.Alice.(Msg2,Encrypt.(SKs,<Alice,Encrypt.
    (SKs,<Alice,PK__.Alice>)>),<>)
env.Alice.(Env0,Bob,<>)
send.Alice.Sam.(Msg1,Alice,<>)
receive.Sam.Alice.(Msg2,Encrypt.(SKs,<Alice,Encrypt.
    (SKs,<Alice,PK__.Alice>)>),<>)
send.Alice.Bob.(Msg3,Encrypt.(SKs,<Alice,PK__.Alice>),<>)
receive.Bob.Alice.(Msg4,Encrypt.(PK__.Alice,<Km>),<Km>)
    leak.Km>
```

The format above is slightly different from the traces shown in [19]. In the latest version of CasperFDR, there are no *fake* and *intercept* channels for the intruder anymore; they are all replaced with the normal *send* and *receive* channels. After exploring the trees in debugger, we can find that the intruder uses *say*, *hear*, and *infer* channels instead. This may help better understand the intruder's behavior, but the attack traces at system level are not clear now because it is hard to tell which message is sent or intercepted by the intruder. Fortunately, the Casper authors provided the *interpret* command, which translates those traces to protocol messages. Using this command in Casper, the traces above are translated to the following message sequence:

```
1. I_Alice ->   Sam   : Alice
2.   Sam   -> I_Alice : {Alice, {Alice, PK(Alice)}{SKs}}
                        {SKs}
0.         ->  Alice  : Bob
1.  Alice  ->  I_Sam  : Alice
2.  I_Sam  ->  Alice  : {Alice, {Alice, PK(Alice)}{SKs}}
                        {SKs}
3.  Alice  ->  I_Bob  : {Alice, PK(Alice)}{SKs}
4.  I_Bob  ->  Alice  : {Km}{PK(Alice)}
  The intruder knows Km
```

From this message sequence, we can illustrate the attack: In the former run, the intruder impersonates a legal user Alice to get a certificate from the server Sam, by the former message 1 and 2. In the following run, in which Alice initiates the protocol, the intruder impersonates the server passing the certificate to Alice[1], by the latter message 1 and 2. Afterwards, when Alice sends the authentication request to Bob by message 3, the intruder will intercept this message and fake his own authentication reply message (message 4), in which he includes the AK generated by himself. Alice will think that the AK is passed by Bob and that it is only known by Bob and her, but in fact, the intruder knows it. This attack happens due to lack of mutual authentication.

The attacks on assertions Secret(B, kb, [A]) and WeakAgreement(B, A) are similar to the attack above, so we skip their traces and analysis due to page limit. When we analyze the attack on assertion WeakAgreement(A, B), we find an attack by FDR. Interpreted the trace from FDR by Casper, we obtain:

```
1. I_Alice ->   Sam   : Alice
2.   Sam   -> I_Alice : {Alice, {Alice, PK(Alice)}{SKs}}
                        {SKs}
3. I_Alice ->   Bob   : {Alice, PK(Alice)}{SKs}
4.   Bob   -> I_Alice : {Kb}{PK(Alice)}
```

[1]We noticed that impersonating the server is not necessary for the attack to succeed, that is, the intruder does not have to get Alice's certificate from server and pass it to her later.

- Message 1. $SS \rightarrow BS : Cert(SS.Manufacture)$
- Message 2. $SS \rightarrow BS$: $N_S \mid Cert(SS) \mid Capabilities \mid BCID$
- Message 3. $BS \rightarrow SS$: $N_S \mid N_B \mid KU_{SS}(AK) \mid SeqNo \mid Lifetime \mid SAIDList \mid Cert(BS) \mid SIG_{BS}(3)$

Fig. 2. Intel Version Authentication Protocol

which means that the intruder could impersonate a legal user, Alice, and communicate with Bob. First, the intruder impersonates Alice to get the certificate from server Sam[2]. Then the intruder can send the authentication request to Bob by message 3, and Bob will finish the run of the protocol with message 4, believing he is communicating with Alice. This corresponds to the Simple Replay Attack proposed in [12], which could be used in a Denial of Service (DoS) attack.

IV. Modeling and Analyzing Intel Nonce version

A. Intel Nonce version of PKM

The Intel Nonce version of PKM authentication protocol is shown in Fig. 2, where N_S and N_B are nonces generated by SS and BS respectively, and $SIG_{BS}(3)$ is the signature of BS over message 3.

B. Modeling the Intel Nonce PKM Protocol in Casper

The Intel version of the PKM protocol can be modeled in Casper as follows[3]:

```
#Protocol description
0.    -> A : B
[A != B]
1.  A -> S : A
2.  S -> A : {A, {A, PK(A) % pka}{sks} % certa}{sks}
3.  A -> B : na, certa % {A, PK(A) % pka}{sks}
[B != A]
4.  B -> S : B
5.  S -> B : {B, {B, PK(B) % pkb}{sks} % certb}{sks}
6a. B -> A : certb % {B, PK(B) % pkb}{sks}
6b. B -> A : {na, nb, {kb}{pka % PK(A)}}{SK(B) % skb}
```

C. Analysis of the Intel version with FDR

After compiling the models with Casper and analyzing the output with FDR, we found one attack on assertion WeakAgreement(A, B). Interpreting the trace from FDR by Casper, we get:

```
 1. I_Alice ->   Sam   : Alice
 2.   Sam   -> I_Alice : {Alice, {Alice, PK(Alice)}
                         {SKs}}{SKs}
 3. I_Alice ->   Bob   : Nm, {Alice, PK(Alice)}{SKs}
 4.   Bob   ->  I_Sam  : Bob
 4.  I_Bob  ->   Sam   : Bob
 5.   Sam   ->  I_Bob  : {Bob, {Bob, PK(Bob)}{SKs}}{SKs}
 5.  I_Sam  ->   Bob   : {Bob, {Bob, PK(Bob)}{SKs}}{SKs}
6a.   Bob   -> I_Alice : {Bob, PK(Bob)}{SKs}
6b.   Bob   -> I_Alice : {Nm, Nb, {Kb}{PK(Alice)}}
                         {SK(Bob)}
```

[2]In fact, the intruder can intercept such information from previous run by Alice instead.

[3]Due to space limit, we only include protocol description part here. Full version of the models can be found in [21].

- Message 1. $SS \rightarrow BS : Cert(SS.Manufacture)$
- Message 2. $SS \rightarrow BS$:
  $N_S \mid Cert(SS) \mid Capabilities \mid BCID$
- Message 3. $BS \rightarrow SS$:
  $N_S \mid N_B \mid KU_{SS}(pre-AK, SSID) \mid SeqNo \mid$
  $Lifetime \mid SAIDList \mid Cert(BS) \mid SIG_{BS}(3)$
- Message 4. $SS \rightarrow BS$:
  $N_B \mid SSAddr \mid AK(N_B, SSAddr)$

Fig. 3. PKMv2 Authentication Protocol

This attack shows that the intruder can impersonate a legal user Alice to send authentication request to Bob, and Bob will respond as if he communicates with Alice. It is similar to the attack shown in the previous section.

V. Modeling and Analyzing PKMv2

A. IEEE 802.16e PKMv2 Protocol

IEEE 802.16e proposes PKMv2, in which one additional message is added at the end of the original protocol, shown as Fig. 3. SSID is SS's identifier from Cert (SS); AAID is the ID of Authorized Association (AA); SSAddr is the MAC address of SS.

B. Modeling PKMv2 in Casper

The Casper model for PKMv2 is similar to the one for Intel Nonce version, except that the Protocol description part has one more message.

```
#Protocol description
0.    -> A : B
[A != B]
1.  A -> S : A
2.  S -> A : {A, {A, PK(A) % pka}{sks} % certa}{sks}
3a. A -> B : certa % {A, PK(A) % pka}{sks}
3b. A -> B : {na, A}{SK(A) % ska}
4.  B -> S : B
5.  S -> B : {B, {B, PK(B) % pkb}{sks} % certb}{sks}
6a. B -> A : certb % {B, PK(B) % pkb}{sks}
6b. B -> A : {na, nb, {kb}{pka % PK(A)}}{SK(B) % skb}
7.  A -> B : {A, nb}{SK(A) % ska}
```

After compiling with Casper and checking with FDR, we did not find any expected attack on this model. We think this is because the intruder is not treated as a legal user and can not get a certificate from the server. Therefore, we removed the server from the specification of the protocol, which in fact, is not included in the original protocol at all. We added the server in previous models because we want to find a way to pass the certificate to the users. Without the server, we have to assume that all the users already know the public keys of all users. The simplified model is shown as follows (with only the protocol description part due to space limit):

```
#Protocol description
0.    -> A : B
[A != B]
1. A -> B : {na, A}{SK(A)}
[B != A]
2. B -> A : {na, nb, {kb}{PK(A)}}{SK(B)}
3. A -> B : {A, nb}{kb}
```

- Message 1. $SS \rightarrow BS : Cert(SS.Manufacture)$
- Message 2. $SS \rightarrow BS$:
  $T_S \mid Cert(SS) \mid Capabilities \mid BCID \mid SIG_{SS}(2)$
- Message 3. $BS \rightarrow SS$:
  $T_S \mid T_B \mid KU_{SS}(AK) \mid SeqNo \mid Lifetime$
  $\mid SAIDList \mid Cert(BS) \mid SIG_{BS}(3)$

Fig. 4. Revised Authentication Protocol with Timestamp

C. Analysis of Simplified PKMv2 Model

After compiling the model above with Casper and checking with FDR, we found an attack on assertion WeakAgreement(A, B). Interpreting the traces from FDR by Casper, we get:

```
0.           ->   Alice   : Mallory
1.   Alice   -> I_Mallory : {Na, Alice}{SK(Alice)}
1.  I_Alice  ->    Bob    : {Na, Alice}{SK(Alice)}
2.    Bob    ->  I_Alice  : {Na, Nb, {Kb}{PK(Alice)}}
                            {SK(Bob)}
2. I_Mallory ->   Alice   : {Na, Nb, {Kb}{PK(Alice)}}
                            {SK(Mallory)}
3.   Alice   -> I_Mallory : {Alice, Nb}{Kb}
3.  I_Alice  ->    Bob    : {Alice, Nb}{Kb}
```

We noticed that this attack is still possible even if we add the server back, as long as the intruder is allowed to get a certificate from the server. This is in fact the spirit of the PKMv2 protocol. But as explained above, we were unable to find a way to model this case in Casper. This attack corresponds to the Interleaving Attack we described in [10]. Note this attack cannot be found by BAN logic.

VI. Modeling and Analyzing our Timestamp Version

A. Our revised PKM Protocol

Our Timestamp version of the PKM authentication protocol is shown in Fig. 4, where T_S and T_B are timestamps generated by the SS and BS respectively; $SIG_{SS}(2)$ is the signature of SS over message 2; and $SIG_{BS}(3)$ is the signature of BS over message 3.

B. Modeling our revised protocol in Casper

Our revised PKM protocol with timestamps can be modeled as follows (with only protocol description part):

```
#Protocol description
0.    -> A : B
[A != B]
1.  A -> S : A
2.  S -> A : {A, {A, PK(A) % pka}{sks} % certa}{sks}
3a. A -> B : certa % {A, PK(A) % pka}{sks}
[B != A]
3b. A -> B : {tsa, A}{SK(A) % ska}
[tsa==now or tsa+1==now]
4.  B -> S : B
5.  S -> B : {B, {B, PK(B) % pkb}{sks} % certb}{sks}
6a. B -> A : certb % {B, PK(B) % pkb}{sks}
6b. B -> A : {tsa, tsb, {kb}{pka % PK(A)}}{SK(B) % skb}
[tsb==now or tsb+1==now]
```

C. Analysis of our protocol with FDR

After compiling and checking with Casper and FDR respectively, we found no attacks like those attacks on PKMv1, Intel Nonce version, and PKMv2. However, we found another attack by FDR, which is interpreted by Casper as follows:

```
 0.         ->   Alice    : Mallory
 1. I_Alice ->    Sam     : Alice
 2.   Sam   ->  I_Alice   : {Alice, {Alice, PK(Alice)}
                             {SKs}}{SKs}
 4.  I_Bob  ->    Sam     : Bob
 1.  Alice  ->   I_Sam    : Alice
 2.  I_Sam  ->   Alice    : {Alice, {Alice, PK(Alice)}
                             {SKs}}{SKs}
3a.  Alice  -> I_Mallory : {Alice, PK(Alice)}{SKs}
3a. I_Alice ->    Bob     : {Alice, PK(Alice)}{SKs}
3b.  Alice  -> I_Mallory : {0, Alice}{SK(Alice)}
3b. I_Alice ->    Bob     : {0, Alice}{SK(Alice)}
 4.   Bob   ->   I_Sam    : Bob
 5.   Sam   ->   I_Bob    : {Bob, {Bob, PK(Bob)}
                             {SKs}}{SKs}
 5.  I_Sam  ->    Bob     : {Bob, {Bob, PK(Bob)}
                             {SKs}}{SKs}
6a.   Bob   ->  I_Alice   : {Bob, PK(Bob)}{SKs}
6b.   Bob   ->  I_Alice   : {0, 0, {Kb}{PK(Alice)}}
                             {SK(Bob)}
```

We noticed that in this attack, the intruder simply replayed immediately the request message sent by Alice (represented by the consecutive message 3a's and 3b's), and Bob granted service to this request by message 6a and 6b. The intruder could not get any benefit from this attack at all. In fact, it has the same effect as when the intruder simply intercepts the last message. Except that effect, the intruder can not do any harm to the system. Usually, this kind of attack is not considered as a vulnerability by the security community [22]. The reason is, if the intruder has the ability to intercept messages, then he can always intercept the last message.

VII. Conclusion and Future Work

In this paper, we modeled the PKM protocols and its later versions in Casper. We compiled those models with Casper and checked the CSP output with FDR. Attacks were found on each version. The attack traces in FDR were interpreted by Casper and the message sequence results are discussed.

During the modeling and analysis, we found several features that we could not implement properly, probably because Casper does not provide necessary functionality. First, we can not assign certificates to users as their initial knowledge. To pass the certificate to users, we have to add a server and let the users request the certificate from the server. This introduced several more messages in the protocol models, which could possibly result in unexpected attacks. Second, the intruder is not treated as a legal user, which is preferred in some cases because some attacks may be performed by legal users. We will explore those features further in our future work.

References

[1] IEEE 802.16 Work Group, "IEEE std 802.16–2001: Air interface for fixed broadband wireless access system," IEEE, 2002.

[2] ——, "IEEE std 802.16–2004: Air interface for fixed broadband wireless access system," IEEE, 2004.

[3] ——, "IEEE std 802.16e–2005: Air interface for fixed broadband wireless access system – amendment: Physical and medium access control layers for combined fixed and mobile operation in licensed bands," IEEE, 2006.

[4] C. A. R. Hoare, "Communicating sequential processes," *Commun. ACM*, vol. 21, no. 8, pp. 666–677, 1978.

[5] C. A. R. Hoare, Ed., *Communicating Sequential Processes*. Prentice Hall International, 1985.

[6] A. W. Roscoe, "Modelling and verifying key-exchange protocols using CSP and FDR," in *Proceedings of 8th IEEE Computer Security Foundations Workshop*, 1995.

[7] Formal Systems (Europe) Ltd, "FDR2 user manual: Failure–divergence refinement," May 2000.

[8] S. Schneider, "Security properties and CSP," in *SP '96: Proceedings of IEEE Symposium on Security and Privacy*, 1996.

[9] G. Lowe, "Casper: A compiler for the analysis of security protocols," *Journal of Computer Security*, vol. 6, pp. 53–84, 1998.

[10] S. Xu and C.-T. Huang, "Attacks on PKM protocols of IEEE 802.16 and its later versions," in *ISWCS '06: Proceedings of the 3rd International Sympsium on Wireless Communication Systems*, Sep. 2006.

[11] D. Johnston and J. Walker, "Overview of IEEE 802.16 security," *IEEE Security & Privacy*, 2004.

[12] S. Xu, M. M. Matthews, and C.-T. Huang, "Security issues in privacy and key management protocols of IEEE 802.16," in *ACMSE '06: Proceedings of the 44th ACM Southeast Conference*, Mar. 2006.

[13] H. Tian, L. Pang, and Y. Wang, "Key management protocol of the IEEE 802.16e," *Wuhan University Journal of Natural Sciences*, vol. 12, no. 1, Jan. 2007.

[14] E. Yuksel, "Analysis of the PKMv2 protocol in IEEE 802.16e–2005 using static analysis," *Informatics and Mathematical Modelling*, 2007.

[15] M. Burrows, M. Abadi, and R. M. Needham, "A logic of authentication," in *Proceedings of the twelfth ACM symposium on Operating systems principles*, 1989.

[16] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," *IEEE transactions on Software Engineering*, 1995.

[17] C. Anson and C. Mitchell, "Security defects in the CCITT recommendation X.509 – the directory authentication framework," *Computer Communication Review*, vol. 20, no. 2, pp. 30–34, Apr. 1990.

[18] CCITT, "CCITT X.509," Internet Draft, Nov. 2002. [Online]. Available: http://www.lsv.ens-cachan.fr/spore/ccittx509_3.pdf

[19] G. Lowe, "Breaking and fixing the Needham-Schroeder public-key protocol using FDR," in *Proceedings of the Second International Workshop on Tools and Algorithms for Construction and Analysis of Systems*, 1996.

[20] B. Donovan, P. Norris, and G. Lowe, "Analyzing a library of security protocols using Casper and FDR," in *Proceedings of Workshop on Formal Methods and Security Protocols*, Jul. 1999.

[21] S. Xu, "Security protocols in WirelessMAN," Ph.D dissertation, University of South Carolina, 2008.

[22] W. Mao, Ed., *Modern Cryptography: Theory and Practice*. Pearson Education, Prentice Hall PTR, 2004.