Bauer-Berson-Feiertag attack revisited*

Jun-Bum Shin[1] and Kwang H. Lee[2]

[1] Softforum and KAIST (jbshin@softforum.com, jbshin@if.kaist.ac.kr)

[2] KAIST (khlee@if.kaist.ac.kr)

Abstract: We show that Shoup and Rubin's protocols are not secure against the BBF attack proposed by Bauer, Berson, and Feiertag, and propose an amendment. Furthermore, our results indicate that both Bellare and Rogaway's security and Paulson's security do not imply the security against the BBF attack.

Keywords: key distribution protocols, attacks

---

1 Introduction

One of the goals of security protocols is to distribute the session key in a secure manner. It is well known that designing a secure protocol is difficult, and a variety of attacks have been proposed (the attacks are nicely summarized in Chapter 10 and 12 of [8]). Among them, we focus on the BBF attack proposed by Bauer, Berson, and Feiertag [1].

The BBF attack is based on the assumption that *a user's long-term key can be compromised*. We say that a protocol is secure against the BBF attack, if it's security can be repaired after the installation of a new key, even though some old keys were compromised. As our long-term keys can be compromised via the system break-ins or a cryptanalysis, the BBF attack is of practical importance.

In this paper, we analyze Shoup and Rubin's protocols, SK1 and SK3 (the smart card version of SK1) [12]. We mention that SK3 is currently implemented [7], and much research has been done analyzing the security of SK1 and SK3: In [12], Shoup and Rubin showed that SK1 satisfies Bellare and Rogaway's security [4], and SK3 satisfies their extended version of that. In [2], Bella showed the correctness of SK3 using his extended version of Paulson's security [11], but in [3], he analyzed SK3 again and showed a weakness of SK3 based on the assumption that an attacker can exploit other users' cards.

We show that both SK1 and SK3 are not secure against the BBF attack and propose an amendment. The problem we found is not recognized in [12, 2, 3], because the models used in their work cannot express the event that a user can install a new long-term key

*This research was partially supported by KOSEF through AITrc.

after discovering the compromise of his/her key. Therefore, our results indicate that there is room for extending formal security definitions based on Bellare and Rogaway's work [4] and Paulson's work [11].

The contributions of this paper are twofold:

(1) We show some weaknesses of Shoup and Rubin's protocols [12], and propose an amendment.

(2) We show that the security definitions proposed by Bellare and Rogaway [4] and Paulson [11] do not imply the security against the important attack proposed by Bauer, Berson, and Feiertag [1].

This paper is structured as follows: In Section 2, we review the BBF attack. We analyze SK1 and SK3 and show that they are not secure against the BBF attack in Section 3 and 4 respectively. Amendments to SK1 and SK3 are proposed in Section 5 and we conclude in Section 6.

2 Preliminary

We review the BBF attack on Needham and Schroeder's symmetric key protocol (NSSK) [1].

2.1 NSSK [9]

We assume that the reader is familiar with NSSK, so we introduce the flows of NSSK only. For more details about NSSK, please refer to [9].

$$A \to S : A.B.N_A \tag{NSSK: 1}$$
$$S \to A : \mathcal{E}_{K_A}(B.N_A.K.\mathcal{E}_{K_B}(A.K)) \tag{NSSK: 2}$$
$$A \to B : \mathcal{E}_{K_B}(A.K) \tag{NSSK: 3}$$
$$B \to A : \mathcal{E}_K(N_B) \tag{NSSK: 4}$$
$$A \to B : \mathcal{E}_K(N_B - 1) \tag{NSSK: 5}$$

where $\mathcal{E}_K(a)$ (resp. $\mathcal{D}_K(b)$) represents the encryption of a (resp. the decryption of b) under key K; $a.b$ represents the concatenation of two bit-strings a and b; and K_A (resp. K_B) represents the long-term key of an initiator Alice (A) (resp. a responder Bob (B)) registered to a key server (S).

2.2 Bauer-Berson-Feiertag attack [1]

Bauer, Berson, and Feiertag have shown that the security of NSSK cannot be repaired when the initiator's long-term key was compromised.

We consider the case that the initiator's old key is compromised, but Alice, the initiator, installs a new key at time τ_0. Then, the following shows that NSSK is not secure against the BBF attack:

1. Alice's old long-term key $K_{A,old}$ is compromised by some means.

2. An attacker impersonating A selects a nonce N_A and sends $A.B.N_A$ to S and intercepts $\mathcal{E}_{K_{A,old}}(B.N_A.K_0.\beta_0)$ from S to A, where $\beta_0 = \mathcal{E}_{K_B}(A.K_0)$

$$I(A) \rightarrow S : A.B.N_A \qquad \text{(NSSK: 1) before } \tau_0$$

$$S \rightarrow I(A) : \mathcal{E}_{K_{A,old}}(B.N_A.K_0.\beta_0) \qquad \text{(NSSK: 2) before } \tau_0$$

   Then, an attacker can get K_0 and $\beta_0 = \mathcal{E}_{K_B}(A.K_0)$, because the attacker knows $K_{A,old}$.

3. The compromise of $K_{A,old}$ is discovered, so she installs a new long-term key $K_{A,new}$ at time τ_0.

4. Some period of time later, an attacker impersonating A sends $\beta_0 = \mathcal{E}_{K_B}(A.K_0)$ to B. Then, after receiving it, B chooses a nonce N_B, and sends $E_{K_0}(N_B)$ to A. An attacker intercepts $E_{K_0}(N_B)$, decrypt it and gets N_B, and sends $E_{K_0}(N_B - 1)$ to B. Note that an attacker can do that because the attacker knows K_0.

$$A \rightarrow B : \beta_0 = \mathcal{E}_{K_B}(A.K_0) \qquad \text{(NSSK: 3) after } \tau_0$$

$$B \rightarrow A : \mathcal{E}_{K_0}(N_B) \qquad \text{(NSSK: 4) after } \tau_0$$

$$A \rightarrow B : \mathcal{E}_{K_0}(N_B - 1) \qquad \text{(NSSK: 5) after } \tau_0$$

   Then, from the protocol logic, B accepts the session key K_0 after receiving $\mathcal{E}_{K_0}(N_B - 1)$. However, an attacker knows K_0.

Potentially, an attacker can see every session key shared between A and B before τ_0, but we hope that NSSK protect the session key after τ_0. However, as NSSK does not satisfy this property, NSSK is not secure against the BBF attack.

Remark 1 *Denning and Sacco have shown another flaw in NSSK: the security of NSSK cannot be repaired when some old session keys were compromised [5].*

3 Analysis of SK1

We review a three-party session key distribution protocol, SK1, proposed by Shoup and Rubin [12], and show that SK1 is not secure against the BBF attack.

3.1 SK1 [12]

We let $a \oplus b$ (resp. $a.b$) be the bit-wise XOR (resp. the concatenation) of two bit-strings a and b, and let $f_K(x)$ be the value of a pseudo-random function for a key K and input x. We assume that an initiator Alice (A) and a responder Bob (B) register their long-term keys, say $\mathbf{K}_A = (K_A, K'_A, K''_A)$ and $\mathbf{K}_B = (K_B, K'_B, K''_B)$, to a key server ($S$). SK1 consists of two phases.

In phase 1, A sends $A.B$ to S; after receiving $A.B$ from A, S sends $\pi.\alpha$ to A, where $\pi = f_{K_B}(A) \oplus f_{K'_A}(B)$ and $\alpha = f_{K''_A}(B.\pi)$; and after receiving $\pi.\alpha$ from S, A rejects if $\alpha \neq f_{K''_A}(B.\pi)$, and otherwise sets $\kappa = \pi \oplus f_{K'_A}(B)$:

$$A \to S : A.B \tag{SK1: 1-1}$$

$$S \to A : \pi = f_{K_B}(A) \oplus f_{K'_A}(B).\alpha = f_{K''_A}(B.\pi) \tag{SK1: 1-2}$$

In phase 2, A chooses r at random and sends it to B; after receiving r from A, B chooses s at random, accepts the session key $sk_B = f_{f_{K_B}(A)}(0.s)$, and sends $s.\beta$ to A, where $\beta = f_{f_{K_B}(A)}(1.r.s)$; and after receiving $s.\beta$ from B, A rejects if $\beta \neq f_\kappa(1.r.s)$, and otherwise accepts the session key $sk_A = f_\kappa(0.s)$:

$$A \to B : r \tag{SK1: 2-1}$$

$$B \to A : s.\beta = f_{f_{K_B}(A)}(1.r.s) \tag{SK1: 2-2}$$

3.2 Attacks

We show that SK1 is not secure against the BBF attack.

Attack-1 on SK1: We consider the case that the initiator's old key is compromised, but Alice, the initiator, installs a new key at time τ_0. Then, the following shows our first attack on SK1:

1. Alice's old long-term key $\mathbf{K}_{A,old} = (K_{A,old}, K'_{A,old}, K''_{A,old})$ is compromised by some means.

2. An attacker impersonating A sends $A.B$ to S and intercepts $\pi_0.\alpha_0$ from S to A, where $\pi_0 = f_{K_B}(A) \oplus f_{K'_{A,old}}(B)$ and $\alpha_0 = f_{K''_{A,old}}(B.\pi_0)$.

$$I(A) \to S : A.B \qquad \text{(SK1: 1-1) before } \tau_0$$

$$S \to I(A) : \pi_0.\alpha_0 \qquad \text{(SK1: 1-2) before } \tau_0$$

3. The compromise of $\mathbf{K}_{A,old}$ is discovered, so she installs a new long-term key $\mathbf{K}_{A,new}$ at time τ_0.

4. Some period of time later, an attacker impersonating A chooses r at random and sends it to B. Then, after receiving r, B chooses s at random, accepts the session key $sk_B = f_{f_{K_B}(A)}(0.s)$, and sends $s.\beta = f_{f_{K_B}(A)}(1.r.s)$ to A. Finally, an attacker intercepts $s.\beta$ from B to A.

$$I(A) \to B : r \qquad \text{(SK1: 2-1) after } \tau_0$$

$$B \to I(A) : s.\beta \qquad \text{(SK1: 2-2) after } \tau_0$$

Note that an attacker can compute $f_{K_B}(A)$ using $\pi_0 = f_{K_B}(A) \oplus f_{K'_{A,old}}(B)$, because the attacker knows $\mathbf{K}_{A,old}$. Therefore, an attacker can compute Bob's session key sk_B.

Attack-2 on SK1: We consider the case that the responder's old key is compromised, but Bob, the responder, installs a new key at time τ_0. Then, the following shows our second attack on SK1:

1. Bob's old key $\mathbf{K}_{B,old} = (K_{B,old}, K'_{B,old}, K''_{B,old})$ is compromised.

2. An attacker impersonating A sends $A.B$ to S and intercepts $\pi_0.\alpha_0$ from S to A, where $\pi_0 = f_{K_{B,old}}(A) \oplus f_{K'_A}(B)$ and $\alpha_0 = f_{K''_A}(B.\pi_0)$.

$$I(A) \to S : A.B \qquad \text{(SK1: 1-1) before } \tau_0$$

$$S \to I(A) : \pi_0.\alpha_0 \qquad \text{(SK1: 1-2) before } \tau_0$$

3. The compromise of $\mathbf{K}_{B,old}$ is discovered, so he installs a new long-term key $\mathbf{K}_{B,new}$ at time τ_0.

4. Some period of time later, an attacker intercepts $A.B$ from A to S, and sends $\pi_0.\alpha_0$ to A.

$$A \to I(S) : A.B \qquad \text{(SK1: 1-1) after } \tau_0$$

$$I(S) \rightarrow A : \pi_0.\alpha_0 \qquad \text{(SK1: 1-2) after } \tau_0$$

Also, an attacker intercepts r from A to B, chooses s at random, and sends $s.\beta_0 = f_{\kappa_0}(1.r.s)$ to A, where $\kappa_0 = f_{K_{B,old}}(A)$. Note that an attacker can do that because the attacker can compute κ_0 using $\mathbf{K}_{B,old}$.

$$A \rightarrow I(B) : r \qquad \text{(SK1: 2-1) after } \tau_0$$

$$I(B) \rightarrow A : s.\beta_0 \qquad \text{(SK1: 2-2) after } \tau_0$$

From the protocol logic, A accepts the session key $sk_A = f_{\kappa_0}(0.s)$ after receiving $\pi_0.\alpha_0$ and $s.\beta_0$. However, an attacker knows sk_A because the attacker knows s and κ_0.

4 Analysis of SK3

We review SK3, the smart card version of SK1, proposed by Shoup and Rubin [12], and show that SK3 is not secure against the BBF attack.

4.1 SK3 [12]

The structure of SK3 is similar to that of SK1, but it defines additional smart card interface for an initiator i and a responder j:

- $T(i)$: a random bit string stored in the smart card of a user i.
- $C_i(1) = (r, f_{T(i)}(r))$, where r is a random number.
- $C_j(2, i, r) = (s, \beta, \omega)$, where s is a random number; $\beta = f_{f_{K_j}(i)}(1.r.s)$; and $\omega = f_{f_{K_j}(i)}(00.s)$.
- $C_i(3, j, r, s, \pi, \alpha, \beta, \gamma) = ()$ or (δ, ω). This is computed as follows. Check if $f_{T(i)}(r) = \gamma$ and $f_{K''_i}(j.\pi) = \alpha$. If not, output $()$. Otherwise, set $\kappa = \pi \oplus f_{K'_i}(j)$. Check if $f_\kappa(1.r.s) = \beta$. If not, output $()$. Otherwise, set $\delta = f_\kappa(01.s)$ and $\omega = f_{f_{K_j}(i)}(00.s)$, and output (δ, ω).
- $C_j(4, i, s, \delta) = 1$ if $f_{f_{K_j}(i)}(00.s) = \delta$, and 0 otherwise.

Other notations are the same as for SK1. Similar to SK1, SK3 consists of two phases.

In phase 1, A sends $A.B$ to S; after receiving $A.B$ from A, S sends $\pi.\alpha$ to A, where $\pi = f_{K_B}(A) \oplus f_{K'_A}(B)$ and $\alpha = f_{K''_A}(B.\pi)$; and after receiving $\pi.\alpha$ from S, A waits for receiving $s.\beta$ from B (SK3: 2-2):

$$A \rightarrow S : A.B \tag{SK3: 1-1}$$

$$S \rightarrow A : \pi = f_{K_B}(A) \oplus f_{K'_A}(B).\alpha = f_{K''_A}(B.\pi) \tag{SK3: 1-2}$$

In phase 2, A sets $(r, \gamma) = C_A(1)$ and sends r to B; after receiving r from A, B sets $(s, \beta, \omega) = C_B(2, A, r)$, and sends $s.\beta$ to A; if A has received both $\pi.\alpha$ from S (SK3: 1-2) and $s.\beta$ from B (SK3: 2-2), A computes $C_A(3, B, r, s, \pi, \alpha, \beta, \gamma)$. If this is (), A rejects, otherwise A assigns this value to (δ, ω), accepts the session key $sk_A = \omega$, and sends δ to B. Finally, after receiving δ from A, B accepts the session key $sk_B = \omega$ if $C_B(4, A, s, \delta) = 1$, and rejects otherwise:

$$A \rightarrow B : r \tag{SK3: 2-1}$$

$$B \rightarrow A : s.\beta = f_{f_{K_B}(A)}(1.r.s) \tag{SK3: 2-2}$$

$$A \rightarrow B : \delta = f_\kappa(01.s) \tag{SK3: 2-3}$$

4.2 Attacks

We show that SK3 is not secure against the BBF attack.

Attack-1 on SK3: We consider the case that the initiator's old key is compromised, but Alice, the initiator, installs a new key at time τ_0. Then, the following shows our first attack on SK3:

1. Alice's old long-term key $\mathbf{K}_{A,old} = (K_{A,old}, K'_{A,old}, K''_{A,old})$ is compromised by some means, i.e., her smart card is lost.

2. An attacker impersonating A sends $A.B$ to S and intercepts $\pi_0.\alpha_0$ from S to A, where $\pi_0 = f_{K_B}(A) \oplus f_{K'_{A,old}}(B)$ and $\alpha_0 = f_{K''_{A,old}}(B.\pi_0)$.

   $$I(A) \rightarrow S : A.B \qquad \text{(SK3: 1-1) before } \tau_0$$

   $$S \rightarrow I(A) : \pi_0.\alpha_0 \qquad \text{(SK3: 1-2) before } \tau_0$$

3. The compromise of $\mathbf{K}_{A,old}$ is discovered, so she installs a new long-term key $\mathbf{K}_{A,new}$ at time τ_0, i.e., she changes her smart card with new key.

4. Some period of time later, an attacker impersonating A chooses r at random and sends it to B. Then, after receiving r from A, B sets $(s, \beta, \omega) = C_B(2, A, r)$, and sends $s.\beta$ to A; an attacker intercepts $s.\beta$ from B to A, computes $\delta_0 = f_{\kappa_0}(01.s)$, where $\kappa_0 = \pi_0 \oplus f_{K'_{A,old}}(B)$, and sends δ_0 to B. Note that an attacker can compute κ_0 because

the attacker knows $\mathbf{K}_{A,old}$.

$$I(A) \to B : r \qquad \text{(SK3: 2-1) after } \tau_0$$

$$B \to I(A) : s.\beta \qquad \text{(SK3: 2-2) after } \tau_0$$

$$I(A) \to B : \delta_0 = f_{\kappa_0}(01.s) \qquad \text{(SK3: 2-3) after } \tau_0$$

From the protocol logic, B accepts the session key $sk_B = f_{\kappa_0}(00.s)$ after receiving δ_0, because $\kappa_0 = \pi_0 \oplus f_{K'_{A,old}}(B) = f_{K_B}(A)$. However, an attacker knows sk_B because the attacker knows s and κ_0.

Attack-2 on SK3: We consider the case that the responder's old key is compromised, but Bob, the responder, installs a new key at time τ_0. Then the security of SK3 cannot be repaired after τ_0 by the same argument as for Attack-2 on SK1. Consider the following sequence of events:

$$I(A) \to S : A.B \qquad \text{(SK3: 1-1) before } \tau_0$$
$$S \to I(A) : \pi_0.\alpha_0 \qquad \text{(SK3: 1-2) before } \tau_0$$

$$A \to I(S) : A.B \qquad \text{(SK3: 1-1) after } \tau_0$$
$$I(S) \to A : \pi_0.\alpha_0 \qquad \text{(SK3: 1-2) after } \tau_0$$
$$A \to I(B) : r \qquad \text{(SK3: 2-1) after } \tau_0$$
$$I(B) \to A : s.\beta_0 \qquad \text{(SK3: 2-2) after } \tau_0$$
$$A \to I(B) : \delta \qquad \text{(SK3: 2-3) after } \tau_0$$

where $\pi_0 = f_{K_{B,old}}(A) \oplus f_{K'_A}(B)$, $\alpha_0 = f_{K''_A}(B.\pi_0)$, $\beta_0 = f_{f_{K_{B,old}}(A)}(1.r.s)$, and $sk_A = f_{f_{K_{B,old}}(A)}(00.s)$.

5 Amendments

The flaws in SK1 and SK3 can be fixed using time-stamps, which are widely used in real-life protocols, including Kerberos [13].

For security issues about time-stamps, please refer to [6, 10]. When it is hard to achieve well-synchronized clocks, we recommend the use of Bellare and Rogaway's protocol, 3PKD [4], instead of our amendment to SK1,

5.1 Amendment to SK1

Our modified version of SK1 (MSK1) is similar to SK1 except for the following: (1) a time-stamp T, which represents A's local clock when A starts the run, is included in every message sent by A; (2) when S and B receive A's message at time τ, they check whether $|T-\tau|<\epsilon$; and (3) for every bit-string a, $f_K(a)$ is transformed to $f_K(a.T)$ (e.g., $sk_B = f_{f_{K_B}(A.T)}(0.s.T)$). The flows of MSK1 are summarized below:

$$A \rightarrow S : A.B.T \qquad \text{(MSK1: 1-1)}$$
$$S \rightarrow A : \pi = f_{K_B}(A.T) \oplus f_{K'_A}(B.T).\alpha = f_{K''_A}(B.\pi.T) \qquad \text{(MSK1: 1-2)}$$
$$A \rightarrow B : r.T \qquad \text{(MSK1: 2-1)}$$
$$B \rightarrow A : s.\beta = f_{f_{K_B}(A.T)}(1.r.s.T) \qquad \text{(MSK1: 2-2)}$$

MSK1 is secure against Attack-1 and Attack-2. Below, τ_i ($i=0,1,2$) represents the real clock, and $\triangle_X$ represents the discrepancy between the real clock and X's local clock. First of all, Attack-1 does not succeed after $\tau_1 = \tau_0 + 2\epsilon + \triangle_S + \triangle_B$, if Alice installs a new key at time τ_0. Suppose that our claim does not hold. Then, the followings are satisfied:

(1) an attacker knows $\pi_0 = f_{K_B}(A.T) \oplus f_{K'_{A,old}}(B.T)$;

(2) Bob accepts the session key after receiving $r.T$ at time τ_2; and

(3) $\tau_1 < \tau_2$.

Note that (1) and (2) imply that $T < \tau_0 + \epsilon + \triangle_S$ and $\tau_2 - \epsilon - \triangle_B < T$ respectively. We, therefore, get $\tau_2 < \tau_0 + 2\epsilon + \triangle_S + \triangle_B = \tau_1$, but that contradicts (3). Secondly, Attack-2 does not succeed after $\tau_1 = \tau_0 + \epsilon + \triangle_S$, if Bob installs a new key at time τ_0, by the same argument as for Attack-1.

5.2 Amendment to SK3

The relation between SK3 and our modified version of SK3 (MSK3) is similar to that between SK1 and MSK1 except for the following:

- If the smart card do not have its clock, (1) T represents the local clock of the initiator's system that has the smart card interface; (2) T is included for the input of $C_i(1)$, $C_j(2,*)$, $C_i(3,*)$, and $C_j(4,*)$; and (3) the responder asks the query to $C_j(2,i,r,T)$ after verifying the validity of T.

- Otherwise, (1) T represents the local clock of the initiator's smart card when it is asked the query $C_i(1)$, and T is additionally included for the output of $C_i(1)$; (2) T is included for the input of $C_j(2,*)$, $C_i(3,*)$, and $C_j(4,*)$ (e.g., $C_j(2,i,r)$ is changed to $C_j(2,i,r,T)$); and (3) the validity of T is checked inside the smart card when it is asked the query $C_j(2,i,r,T)$.

Finally, in any cases, we change $f_K(a)$ in SK3 to $f_K(a.T)$ in MSK3.

Note that the flows of our two amendments are same, and they are summarized below:

$$A \to S : A.B.T \tag{MSK3: 1-1}$$
$$S \to A : \pi = f_{K_B}(A.T) \oplus f_{K'_A}(B.T).\alpha = f_{K''_A}(B.\pi.T) \tag{MSK3: 1-2}$$
$$A \to B : r.T \tag{MSK3: 2-1}$$
$$B \to A : s.\beta = f_{f_{K_B}(A.T)}(1.r.s.T) \tag{MSK3: 2-2}$$
$$A \to B : \delta = f_\kappa(01.s.T) \tag{MSK3: 2-3}$$

MSK3 is secure against Attack-1 and Attack-2, by the same argument as for MSK1.

6 Conclusion

In this paper, we showed some weaknesses of Shoup and Rubin's protocols, and proposed an amendment. Additionally, our results indicate that there is room for extending previous security notions based on Bellare and Rogaway's work [4, 12] and Paulson's work [11, 2, 3], because they do not imply the security against the important attack proposed by Bauer, Berson, and Feiertag [1].

References

[1] R.K. Bauer, T.A. Berson, and R.J. Feiertag. A key distribution protocol using event markers. *ACM Transactions on Computer Systems*, 1(3):249–255, 1983.

[2] G. Bella. Modelling Security Protocols Based on Smart Cards. In *Proc. of CrypTEC'99, International Workshop on Cryptographic Techniques and E-Commerce, July1999, Hong Kong, China*, 1999.

[3] G. Bella. Lack of explicitness strikes back. In *Security Protocols Workshop*, pages 87–93, 2000.

[4] M. Bellare and P. Rogaway. Provably secure session key distribution . the three party case. In *Annual Symposium on the Theory of Computing (STOC)*, 1995.

[5] D. Denning and G. Sacco. Timestamps in Key Distribution Protocols. *Communications of the ACM*, 24(8):533–536, 1981.

[6] L. Gong. A security risk of depending on synchronized clocks. *Operating Systems Review*, 26(1):49–53, 1992.

[7] R. Jerdonek, P. Honeyman, K. Coffman, J. Rees, and K. Wheeler. Implementation of a Provably Secure Smartcard-based Key Distribution Protocol. In *Proceedings of Third Smart Card Research and Advanced Application Conference*, 1998.

[8] A. J. Menezex, P. C. Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997.

[9] R. Needham and M. Schroeder. Using Encryption for Authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[10] B. C. Neuman and S. G. Stubblebine. A note on the Use of Timestamps as Nonces. *Operating Systems Review*, 27(2):10–14, 1993.

[11] L C Paulson. The inductive approach to verifying cryptographic protocols. *J. Computer Security*, 6:85–128, 1998.

[12] V. Shoup and A. Rubin. Session-key distribution using smart cards. In *Eurocrypt '96*, 1996.

[13] J. G. Steiner, B. C. Neuman, and J. I. Schiller. Kerberos: An authentication service for open network systems. In *Proceedings of the Winter 1988 Usenix Conference*, pages 191 – 201, 1988.