Chapter 7

A Proof of Revised Yahalom Protocol

Although the Yahalom protocol, proposed by Burrows, Abadi, and Needham in 1990 [10], is one of the most prominent key establishment protocols analyzed by researchers from the computer security community (using automated proof tools), the protocol does not possess a security proof within a computational complexity framework. We note that in a recent work of Backes and Pfitzmann [2], a simplified version of this protocol is proven in the cryptographic library that corresponds to a slightly extended Dolev–Yao model [15].

In this chapter, we present a protocol for key establishment that is closely based on the Yahalom protocol. We then present a security proof in the Bellare–Rogaway model and the random oracle model. We hope that by providing such a proof for a slightly modified Yahalom protocol, this will be of interest to the researchers, in particular to researchers from the computer security community. We also observe that no partnering mechanism is specified within the Yahalom protocol (as in the case for many other key establishment protocols). In a real world setting, it is normal to assume that a host can establish several concurrent sessions with many different parties. Sessions are specific to both the communicating parties. In the case of key distribution protocols, sessions are specific to both the initiator and the responder principals, where every session is associated with a unique session key. Session identifiers (SIDs) enable unique identification of the individual sessions. We then present a brief discussion on the role and the possible construct of session identifiers as a form of partnering mechanism, which allows the right session key to be identified in concurrent protocol executions. We recommend that SIDs should be included within protocol specification rather than consider SIDs as artefacts in protocol proof.

Material presented in this chapter has appeared in the following publication:

7.1 The Yahalom Protocol and its Simplified Version

We now revisit the Yahalom protocol [10] described in Protocol 7.1. At the end of Protocol 7.1’s execution, both users $A$ and $B$ will accept the session key ($SK_{AB}$) generated by the trusted server, $S$. Other notation in Protocol 7.1 is as follows: $E(m)_K$ denotes an encryption of some message $m$ under symmetric key $K$; $S$ denotes a server who shares long-term symmetric keys $K_{AS}$ and $K_{BS}$ with $A$ and $B$ respectively; $N_A$ and $N_B$ denote nonces generated by $A$ and $B$ respectively.

1. $A \rightarrow B : N_A$
2. $B \rightarrow S : B, E(A,N_A,N_B)_{K_{BS}}$
3. $S \rightarrow A : E(B,SK_{AB},N_A,N_B)_{K_{AS}}, E(A,SK_{AB})_{K_{BS}}$
4. $A \rightarrow B : E(A,SK_{AB})_{K_{BS}}, E(N_B)_{SK_{AB}}$

**Protocol 7.1:** The Yahalom protocol

Protocol 7.1 provides key confirmation — $B$ is assured that $A$ actually has possession of the same secret session key, $SK_{AB}$, since $A$ sends to $B$ the encryption of the nonce chosen by $B$, $N_B$, using $SK_{AB}$.

Choo and Hitchcock [13] pointed out informally that it does not appear possible to prove Protocol 7.1 secure in the BR93 model due to the encryption of the nonce using the established session key (i.e., $E(N_B)_{SK_{AB}}$) in the last message (from $A$ to $B$). In an independent yet related work, Backes and Pfitzmann [2] raise similar observation. In the simplified version proposed by Backes and Pfitzmann [2], the encryption of the nonce using the established session key (i.e., $E(N_B)_{SK_{AB}}$) in message 4 is removed from the protocol for the following reason.

- Recall that security in the BR93 model is defined using a game simulation, $G$, played between the adversary, $A$, and a collection of player oracles, as described in Chapter 2.3.3 and success of $A$ in $G$ is quantified in terms of $A$’s advantage in distinguishing whether $A$ receives a real key or a random value from the game simulator.
- In the context of the proof simulation for Protocol 7.1, $A$ can perform the follow set of actions.

Stage 1. Asks a series of Send queries that model the above simulation of Protocol 7.1. For example, the adversary, $A$, obtains nonce $N_A$ after asking a Send($A,B,*$) query. $A$ then proceed to choose nonce $N_B$ and ask a Send($B,A,(N_A,N_B)$) query where the game simulator will respond as per protocol specification.

Stage 2. Decides that this particular session is the Test session, and then asks a Test query. The game simulator returns the key, $SK_b$.