CS 395T - Design and Analysis of Security Protocols
Fall 2004

Homework #1

Due: 3:30pm CDT, September 20, 2004

Overview

The purpose of this assignment is to get you started with the Murφ verifier. In the first problem, you are required to extend the provided Murφ model of the Needham-Schroeder public-key protocol with a new invariant. In the second problem, you are required to construct a Murφ model for a simple key distribution protocol (you may use the Needham-Schroeder model as a template). For both problems, you need to execute the model and interpret the results of the verification.

Collaboration policy

No collaboration is permitted on this assignment. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at `http://www.cs.utexas.edu/users/ear/CodeOfConduct.html`

Submission

Submission deadline is 3:30pm CDT, Monday, September 20, 2004.

Combine your Murφ models and text descriptions of the results (*e.g.*, text files containing Murφ printouts of attack traces) into a single `tar` file, and send it as an email attachment to `shmat@cs.utexas.edu` with *CS 395T hw#1* in the subject line.

Running Murφ

Murφ has been installed in `/projects/shmat/Murphi3.1/`
The Murφ model of the Needham-Schroeder public-key protocol is in `/projects/shmat/Murphi3.1/ex/secur/ns.m`
Murφ tutorial and user manual are in `/projects/shmat/Murphi3.1/doc`

Homework

Problem 1 (5 points)

The Needham-Schroeder public-key protocol described in Lecture 1 can be viewed as a key establishment protocol. At the end of the protocol execution, A and B can `xor` their respective nonces together, and use $N_a \oplus N_b$ as the shared symmetric key.

Write a Murφ invariant specifying secrecy of the shared key, add it to the provided model of the Needham-Schroeder protocol, and verify whether it holds.

Hint: the intruder will know $N_a \oplus N_b$ if and only if he learns both N_a and N_b.

Problem 2

The Yahalom protocol is a 3-party key establishment and authentication protocol between Alice (A), Bob (B) and a trusted Server (S). In the beginning of the protocol, A and S share the symmetric key K_{as}, B and S share the symmetric key K_{bs}. The objective of the protocol is to (i) create a fresh symmetric key K_{ab} between A and B, and (ii) authenticate A to B. The protocol proceeds as follows:

$$
\begin{array}{lllll}
1. & A & \rightarrow & B & A,\ N_a \\
2. & B & \rightarrow & S & B,\ \{A, N_b, N_a\}_{K_{bs}} \\
3. & S & \rightarrow & A & \{B, K_{ab}, N_a, N_b\}_{K_{as}},\ \{A, K_{ab}\}_{K_{bs}} \\
4. & A & \rightarrow & B & \{A, K_{ab}\}_{K_{bs}},\ \{N_b\}_{K_{ab}}
\end{array}
$$

A starts the protocol by generating a fresh random number (nonce) N_a, and sends N_a to B together with A's identity. B generates his own nonce N_b, creates a message consisting of A's identity and the two nonces, encrypts it with the symmetric key K_{bs} and sends it to S together with his identity. S decrypts the message and learns A's identity and the nonces. S then generates a fresh symmetric key K_{ab} for A and B, and sends a two-part message to A. In the first part, the new key K_{ab} is encrypted together with B's identity and the nonces by the symmetric key K_{as}, which is shared between S and A. In the second part, K_{ab} is encrypted together with A's identity by the symmetric key K_{bs}, which is shared between S and B (but not A). Finally, A decrypts the first part of the message from S and learns K_{ab}. A then forwards the second part (which he cannot decrypt) to B. To prove that A successfully decrypted the first part and learned the new key K_{ab}, A also sends B's own nonce back to B encrypted with K_{ab}.

Problem 2a (15 points)

The first objective of the Yahalom protocol is *secrecy.* No party other than A, B, or S should be able to learn the new key K_{ab}.

Write a Murφ model of the Yahalom protocol, specify the secrecy property for K_{ab} as a state invariant, and verify whether it holds for every reachable state of your finite-state model.

Hint: consider more than one concurrent session of the protocol, and investigate intruder rules that ignore message formats (*e.g.*, the attacker may put two nonces in a message even if only one nonce is expected by the recipient).

Problem 2b (5 points)

The second objective of the Yahalom protocol is *authentication.* If B thinks that he is talking to A after the protocol has completed, then it should always be the case that he is indeed talking to A.

Specify the authentication property as a state invariant, add it to the model you implemented for Problem 2a, and verify whether it holds for the Yahalom protocol.