Frederick Akalin, Siu Hong Yuen
CS259 Project Writeup

Analysis of the SILC Protocol using Murphi

Abstract
SILC (Secure Internet Live Conferencing) is a protocol designed to provide secure chat and messaging services by encrypting and authenticating all packets. The actual protocol encompasses many forms of conferencing, but our analysis of the SILC protocol focuses on the part in the SILC protocol that allows clients in the same channel to talk to each other securely. We wrote up a Murphi model of the protocol and wanted to see if a malicious client can "eavesdrop" on a conversation on a channel to which he does not belong.

Overview of SILC protocol

When a client wants to join a channel, it first establishes a secure connection with the server via the secure key exchanges using a variation of the Diffie-Hellman protocol. Upon establishing the session key, the client can then send a join command to the server who will then generate a channel key and broadcasts this new channel key to all the clients in the channel. Note that only the packet header is encrypted with the session key, whereas the payload of the packet is encrypted with the channel key.

Similarly, when a client wants to part a channel, it sends a part command to the server, and the server, upon receiving the part command, will generate a new channel key and broadcast it to the other clients still in the channel. Each message sent by a client to the channel is broadcast to all members of the channel encrypted with the current channel key.

Our Murphi model

Our intruder model in Murphi does not have many capabilities. Since we have assumed perfect key exchange between the client and the server, the intruder cannot decrypt any packets in the network. However, it can intercept packets and store them, forward stored packets, as well as work with a malicious client or server by passing packets to the malicious client or server to decrypt.

Results

While doing rational reconstruction, we tried to remove the feature of the SILC protocol which generates a new channel key every time a client joins or parts a channel. Murphi found an exploit in the modified protocol where the intruder joins the channel, parts and is still able to decrypt packets sent thereafter even though it has already parted the channel.

Another exploit that our code found out is that assuming Bob joins #foo, and Murphy joins #foo. Both Bob and Murphy have key K1 at this point. Murphy parts #foo and the server tries to send key K2 to Bob. The intruder blocks the message, and from then on if Bob sends messages encrypted with K1, Murphy will still be able to decrypt it. (More details of the exploit is available in the slides)

Running our Murphi Code

There are a few switches in silcTalk.m that control intruder behavior. For rational reconstruction, all switches should be off except for MAL_CLIENTS. For the actual exploit, turn on REGENERATE_CLIENT_KEY.

The compiled C program should be run with the switches “-ndl –tv –vdfs –m50”. Note that if breadth-first search is used, neither exploit will be found.