Blakely Madden & Jacob Brandt

Prof. Guevara Noubir

Network Security

November 6, 2015

Problem Set 4

Protocol Overview

- Clients either provide a public/private key pair, or one is generated for them on start up
- Clients authenticate to a server with a password over SRP
- Clients send the server their public key
- The server stores the clients' public keys, usernames, and IP addresses
- Client A makes a request to speak with client B
- The server gives A all of B's stored information
- The clients use asymmetric encryption to directly establish a session key
- The clients chat using symmetric encryption and HMAC for authentication

Protocol Details

Clients use SRP to authenticate with the server and become available for chatting

SRP notation:

- g: a generatior of the multiplicative group
- k: a parameter derived by both sides
- s: small salt
- p: user password
- v: server password verifier: $v = g^x$ where at a minimum x = Hash(s,p)
- A and B: random one time ephemeral keys of the user and host, respectively

SRP protocol breakdown:

- Bob → Server: I am Bob, $\{x = Hash(s, p)\}_{PK\_Server}$, and s
- Server stores $v = g^{x}$ and s and Identity of Bob
- Bob → Server: I am Bob and $A = g^{a}$
- Server → Bob: s and $B = kv + g^{b}$
- Both: $u = Hash(A, B)$
- Bob: $S_{Bob} = (B - kg^{x})^{(a + ux)} = (kv + gb - kg^{x})^{(a + ux)} = (kg^{x} - kg^{x} + g^{b})^{(a + ux)} = (g^{b})^{(a + ux)}$
- Bob: $K_{Bob} = Hash(S_{Bob})$
- Server: $S_{Server} = (Av^{u})^{b} = (g^{a}v^{u})^{b} = (g^{a}(g^{x})^{u})^{b} = (g^{a} + u^{x})b = (g^{b})^{(a + ux)}$
- Server: $K_{Server} = Hash(S_{Server}) = K_{Bob}$
- Bob → Server: $M_1 = H(A \mid B \mid S_{Bob})$. Server verifies M_1
- Server → Bob: $M_2 = H(A \mid M_1 \mid S_{Server})$. Bob verifies M_2

After SRP:

- Bob → Server: $\{PKBob\}_{K\_Bob}$ (over tcp)
- Bob → Server: $\{\text{Id of Alice}\}_{K\_Bob}$
- Server → Bob: $\{\text{Location of Alice} \mid PKAlice\}_{K\_Bob}$

Client session key establishment and messaging:

Alice repeats same steps as Bob with the server

1. Bob → Alice: $[\{K\}_{PK\_Alice}]_{PK\_Bob}$ **Update:** Alice no longer participates in key genration
1. Bob → Alice: $[\{(Message_1\}_K]_{HMAC}$

2. Alice → Bob: $[\{Message_2\}_K]_{HMAC}$

Issues

DoS: In order to avoid DoS attacks, the server will temporarily lock multiple failed authentication attempts.

Weak Passwords: SRP protects against attacks related to weak user passwords

Perfect Forward Secrecy: Clients establish symmetric session keys, for each new messaging session, using public key encryption. Clients forget these keys after each session

Malicious Server: A malicious server is still capable of performing man in the middle attacks to recover information exchanged between clients. A trusted server is a requirement of the protocol.

Summary

This protocol is designed to provide perfect forward secrecy and protect against weak passwords and DoS attacks. Without prior knowledge of some encryption information, it is virtually impossible for clients to establish a secure session on their own. For this reason, we use a server to exchange public key and location information between two clients. Unfortunately, this also makes our protocol vulnerable to information forgery and man in the middle attacks by the trusted server.

The protocol is designed to be fast (using symmetric encryption after initial key establishment) and secure (given a trusted server).