SIMPL

Secure Instant Messaging Protocol of Lukes
(Post feedback session)

SIMPL

- TCP-based
- Stages
  - Login
  - Discovery
  - Negotiate
  - Chat
  - Leave
  - Logout

Git Hub

https://github.com/syreal17/SIMPL

Assumption

1. Clients have the server’s public key

Passwords

- Enforced client-side
- Requirements (Not implemented)
  - One (1) or more numerical digit
  - One (1) or more symbol
  - One (1) or more alphabetic uppercase
  - One (1) or more alphabetic lowercase
  - Length 20 or greater
    - XKCD CorrectHorseBatteryStaple

SIMPL Login

Client | Server

Login →

← $[h(R_1+R_2),\ R_1]_{Ks}$

$R_2,\ \{username, W_1, N\}_{Ks}$ →

← (Ok | Deny)

Puzzle prevents DOS to server.
Signing prevents DOS to client.

If the server does not already know the username sent to it, it stores the username mapped with the attached password hash.

where R_1 = large random number
where R_2 = variable length random number
where W_1 = h(Client1 password)
where N = random nonce
where Server stores $K_1 = h(W_1, N)$ for duration of session

SIMPL Discovery

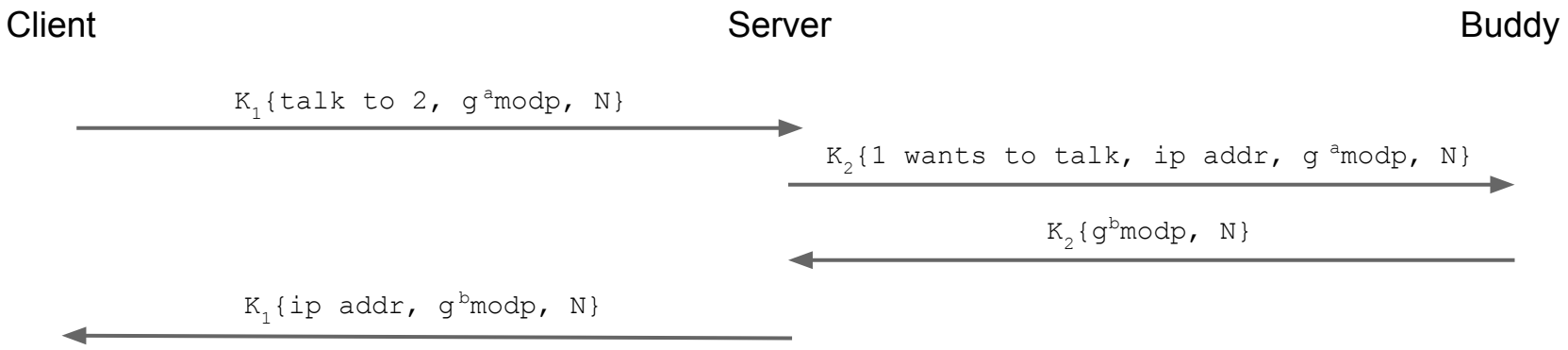
Client Server

$K_1\{Discover\}$ →

← $K_1\{L\}$

where L = list of all logged in clients

SIMPL Negotiate



Clients forget a & b as soon as they have created the shared key.
Buddy is a peer SIMPL Client

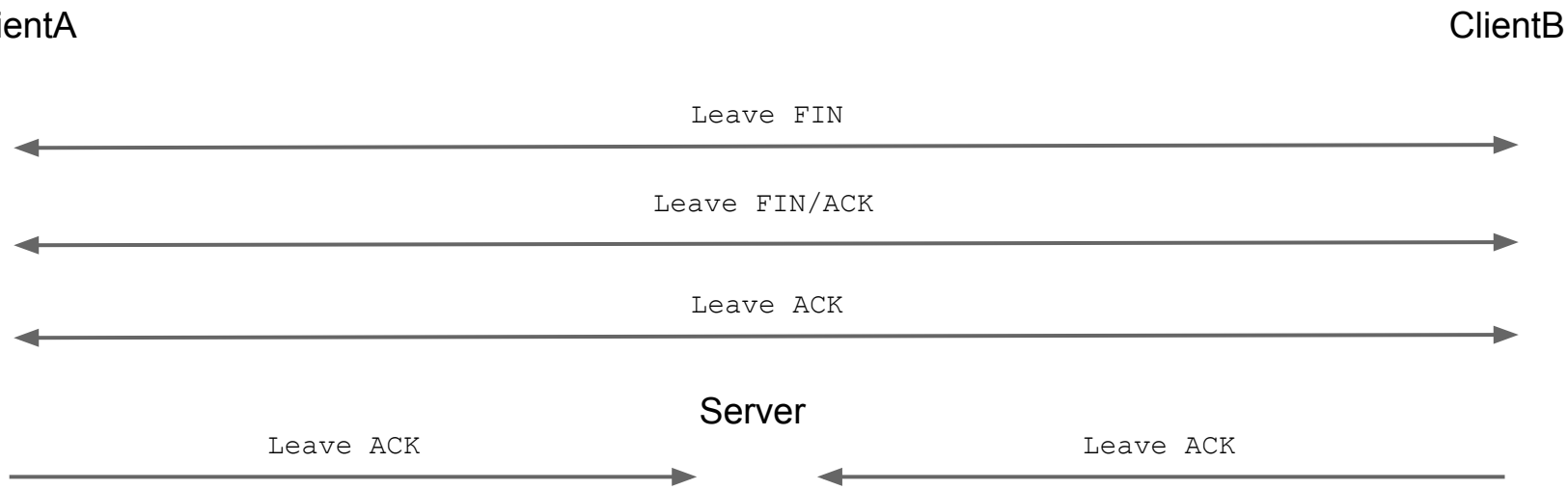
SIMPL Chat

ClientA

ClientB

g^{ab}modp{messages}

SIMPL Leave



Clients must forget $g^{ab} \bmod p$

SIMPL Logout

Client | Server

Logout FIN →

← Logout FIN/ACK

Logout ACK →

Server must forget K_1

Clients must forget $g^{ab} \bmod p$, if they haven't yet

Client1 forgets K_1 after it transmits the final ACK. If Server does not receive the ACK after some timeout t, it logs out Client1 and forgets K_1 anyway.

3-way handshake similar to TCP to prevent either Client or Server from forgetting K_1 before Client1 is successfully logged out.

Services SIMPL Provides

- Communication Confidentiality, even from Server
- Integrity
- Availability, prevents DOS
- Mutual Authenticity
- Perfect Forward Secrecy

Services SIMPL Doesn’t Provide

- End-point Hiding

Implementation Notes

- Have Debug and Release builds.
  - Debug
    - parts of protocol can be turned off
      - good for practicing exploitation
  - Release
    - all parts of protocol must be enabled
      - ensure that we don’t accidentally use a weak version of SIMPL in live test
- Encrypting Server db, though useless long-term against a skilled adversary, might be enough of a thwart for the live test to avoid compromise

Implementation Notes

- Rather than a different packet type for every message, have types: ClientServerPreSession, ClientServerSession, ClientClientSession.
  - These are the only kinds of encryption that will be dealt with
- Have the standard packet format include flags about what information is included or what the packet intention is, thereby the specific message can be determined.
- Perhaps flags should implicitly be outside any crypto notation

Questions?

- Should Leave and Logout not be encrypted?
  - Is the identity of all parties assured enough to forgo encrypting a packet for just containing flags?