Login/ Authentication Protocol

✓A → S: K_s\{A\}

K_s is servers master key which is shared between all clients.

✓S → A: K_A \{Puzzle\}

✓A → S: K_A \{answer, g^amod p , R1\}

K_A is derived from username and password of A

K_{AS} = g^{as}mod p used for communication between the client and server

✓S → A: K_A \{ g^smod p\} , K_{AS}\{salt,R1+1\}

✓A → S: K_{AS} \{ hash(salt|password), R1+2 \}

✓S → A: K_{AS} \{ ACK/RST, R1+3 \}

Client to server

For any communication between the client A and server, we would use K_{AS}

For any request by the client the server will either send a response or will ack it

List

✓A → S: K_{AS} { list || R(previouse nouce+1) }

✓S → A: K_{AS}{user list, R(previouse nouce+1)}

Logout

✓A → S: K_{AS} { logout || R(previouse nouce+1) }

✓S → A: K_{AS}{ ACK, R(previouse nouce+1)}

Initiating a client lookup

To communicate with other clients

✓A → S: K_{AS} { B || R(previouse nouce+1) }

✓S → A: K_{AS}{R(previouse nouce+1) , K_{AB} ,time_to_live, identity of B, ticket_to_B}

Identity of B will have username, IP, Port of B

✓ticket_to_B = K_{B}{identity of A, K_{AB} , B, time_to_live}

Identity of A will have username, IP, Port of A

Key Establishment and Message Exchange

- ✓A → B : K_{AB} { g^X mod p , R1}, ticket_to_B
- ✓B → A : K_{AB} { g^Y mod p , R1+1}

K = g^{XY} mod p generated on both sides

- ✓A → B: K{N1, message}, Hash(message)
- ✓B → A: K{N1, message}, Hash(message)