Secure Chat Application Design

Sourabh Bollapragada
Shardul Thakre

Contents

- Notation
- Features
- Architecture
- Login
- Message Sending and Receiving
- List
- Logout
- Heartbeat
- Key Generation and Password Hashing
- Packet Format

Notation

- A,B = Clients
- S = Server of the system
- PUZ = The Puzzle given by Server.
- SOL = Solution to Puzzle
- M = message
- N's = Nonces
- t = timestamp
- P = Public Key

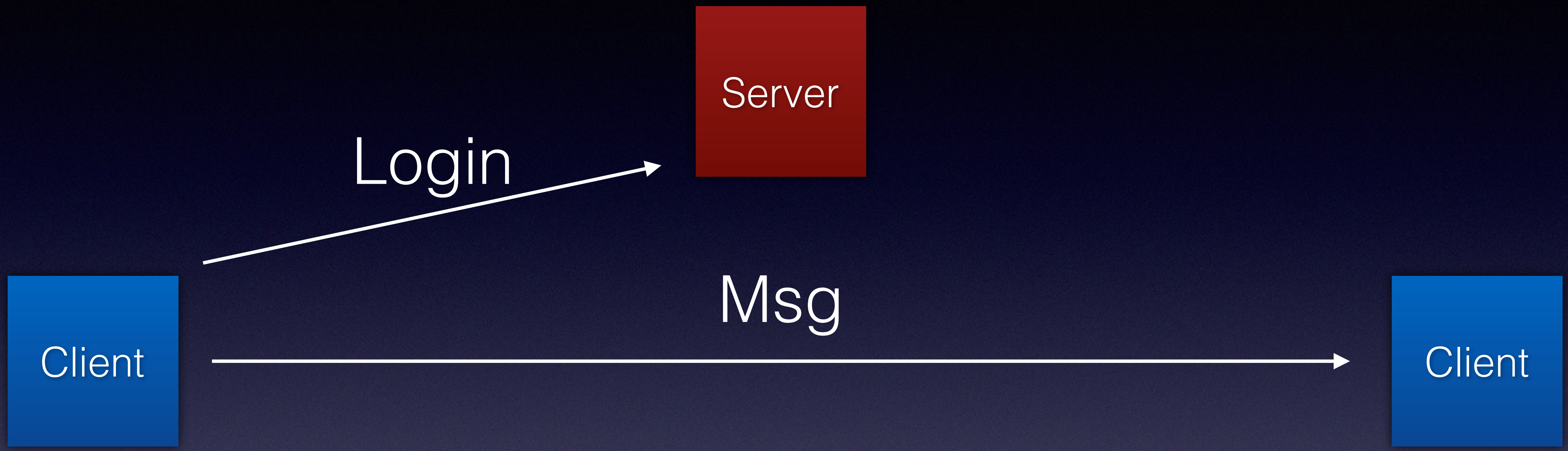
Features

- Authentication
- Confidentiality
- Integrity
- Non Repudiation
- Identity Hiding
- DoS Resistance
- Perfect Forward Secrecy
- Password Cracking Resistance
- End - End Encryption

Changes

- Added an extra assumption
- Added an extra accept/reject message in login which should be obvious, but forgot to add last time.
- Added some more additional details like broadcasting logout by server which was also missed last time.
- Added Diffie Hellman Algorithm in Crypto

Architecture



- We have a Central Server that handles logins of clients.
- The Clients Communicate with each other directly.

Crypto Algorithms

- Symmetric Encryption = AES-GCM 256 bit
- RSA Key Size = 2048 bits
- Diffie Hellman = Elliptic Curve SECP256R1
- Password Storage = PBKDF2
- KDF = PBKDF2 with SHA 512

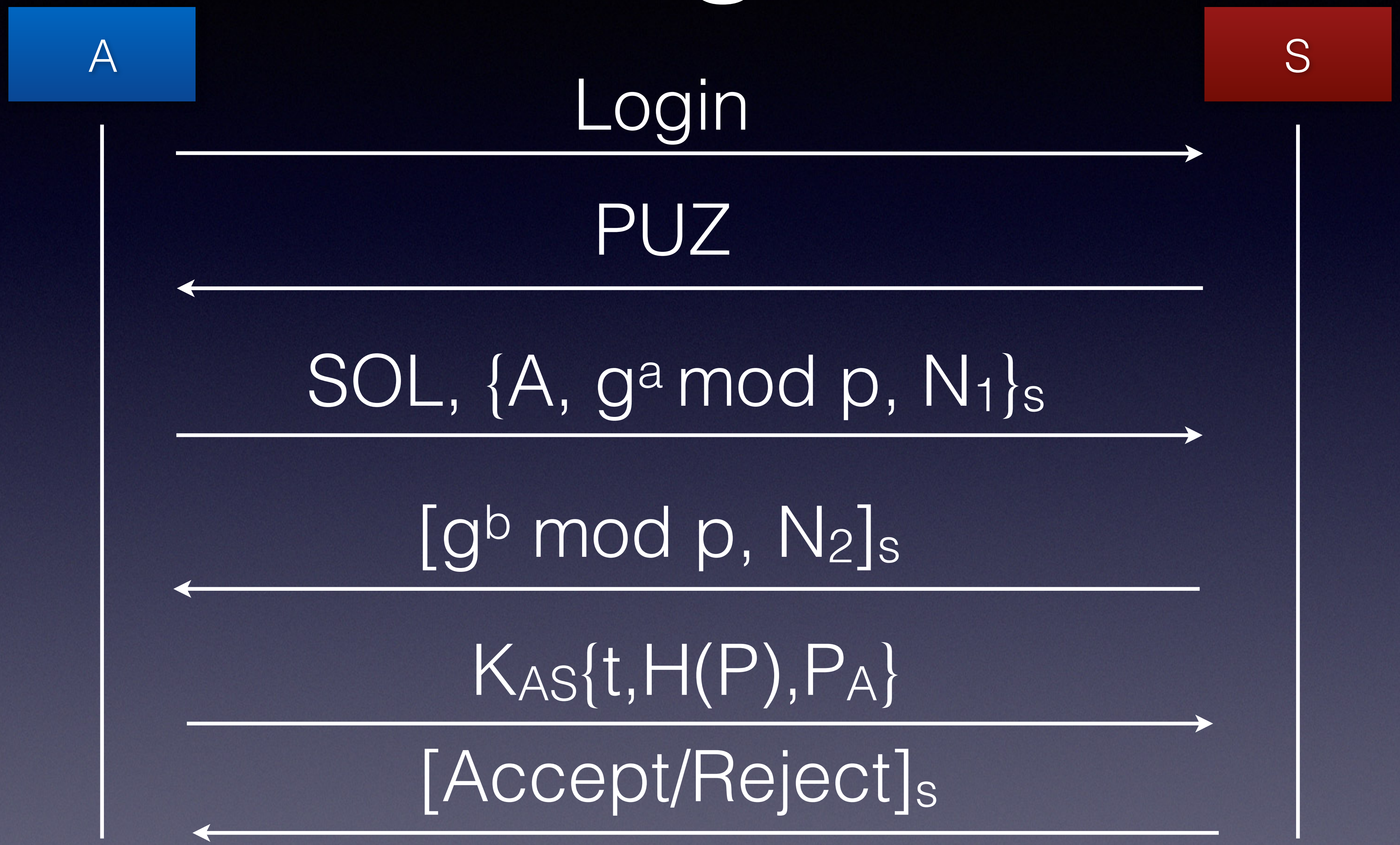
Assumptions

- Each Client has the correct public key of Server.
- The clocks of all clients and server is synchronized and this synchronization cannot be interrupted by adversary.
- The Server is trusted by all clients after the login.
- The Logout Broadcast Reaches Clients.

Important Info

- Every Packet has a Timestamp that the Server or Client uses to prevent replay.
- Most of the Packets have a Signature that is calculated by using the timestamp and other fields.
- All Sockets are UDP
- Active Clients send Heartbeats every 30s.

Login



A

S

Login

PUZ

SOL, $\{A, g^{a} \bmod p, N_1\}_s$

$[g^{b} \bmod p, N_2]_s$

$K_{AS}\{t,H(P),P_A\}$

$[\text{Accept/Reject}]_s$

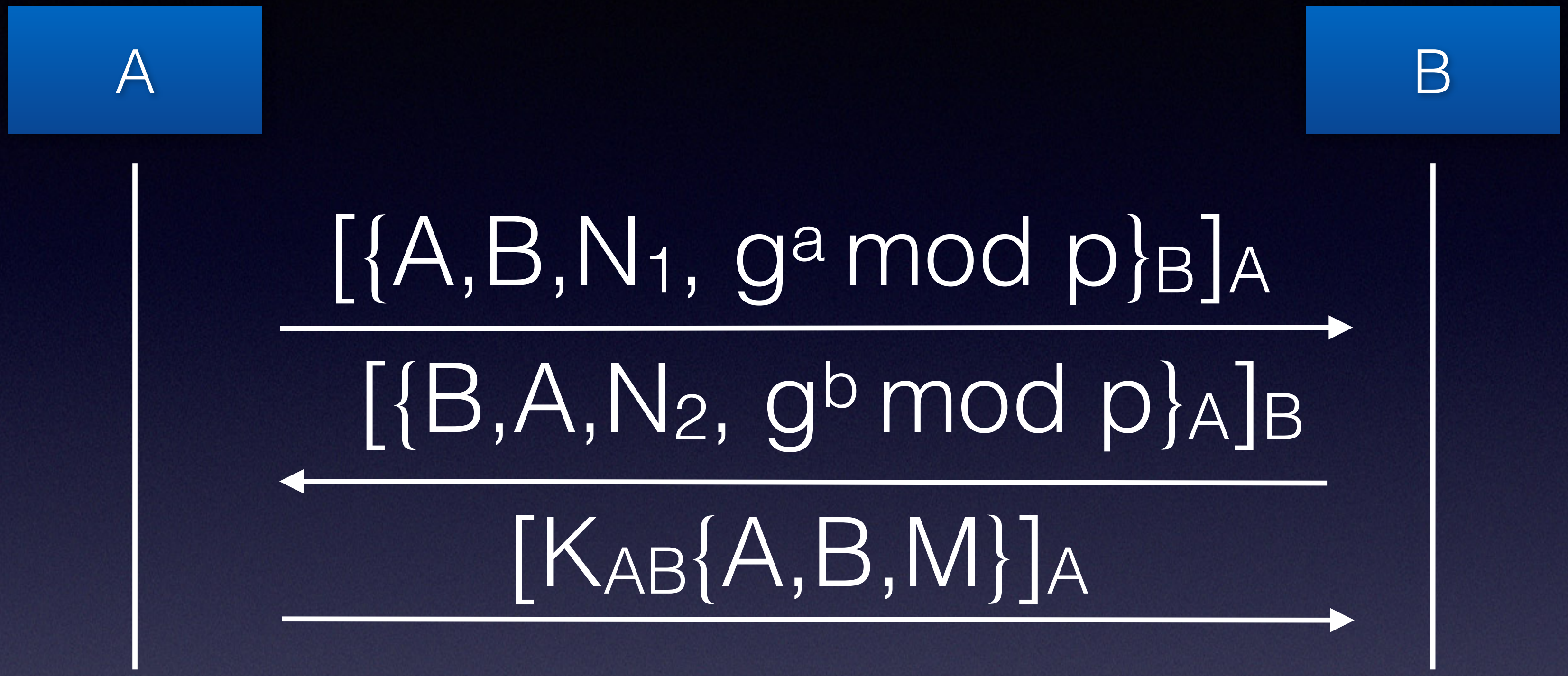
Login contd...

- H(P) is the output of KDF in Password Hashing.
- N_1 and N_2 are used for K_{AS} generation.
- t should be equal to timestamp in last packet.

Client Puzzle

- The Server prepares a Puzzle Certificate with $[t,D,N_S]_S$
  - t is the expiry timestamp of certificate.
  - D is the difficulty of the puzzle
  - N_S is the Server's Nonce.
- When Client requests to login, the server sends the certificate which is precomputed for D and N_S
- Then the Client verifies the certificate and computes $H(N_S, N_C, X)$ where N_C is the client Nonce and X is the solution. The client sends N_C, X, $H(N_S, N_C, X)$ to the Server.
- The Server checks if N_C is not repeat for the current N_S and then computes $H(N_S, N_C, X)$, if the first D bytes are 0 then the solution is correct and the Server continues with Authentication else it rejects it.
- N_S is changed after a certain amount of time after which the certificate is recomputed. If the Server is attacked then the certificate is remade for every change to D.
- When N_S is changed then all N_C's which were recorded will be dropped.
- We modified the puzzle from this paper http://www.tcs.hut.fi/old/papers/aura/aura-nikander-leiwo-protocols00.ps

Message



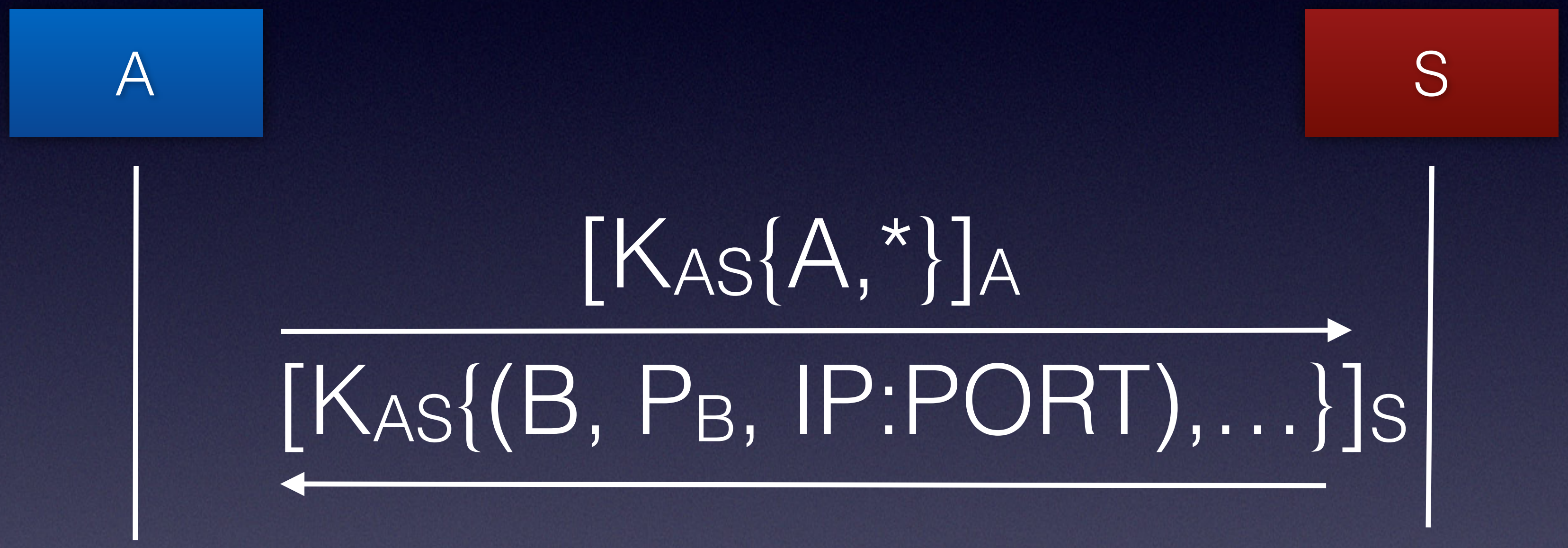
$[\{A,B,N_1, g^a \bmod p\}_B]_A$

$[\{B,A,N_2, g^b \bmod p\}_A]_B$

$[K_{AB}\{A,B,M\}]_A$

- A sends N1 and its DH contribution which is signed by A and encrypted using P_B
- B sends back similar packet if initial packet is authentic.
- K_{AB} is derived from DH, N_1 and N_2 using Key generation algorithm.
- We will use timestamps or sequence numbers to keep track of duplicates or replays.
- Accept is sent by B is message is received successfully from A

List



A

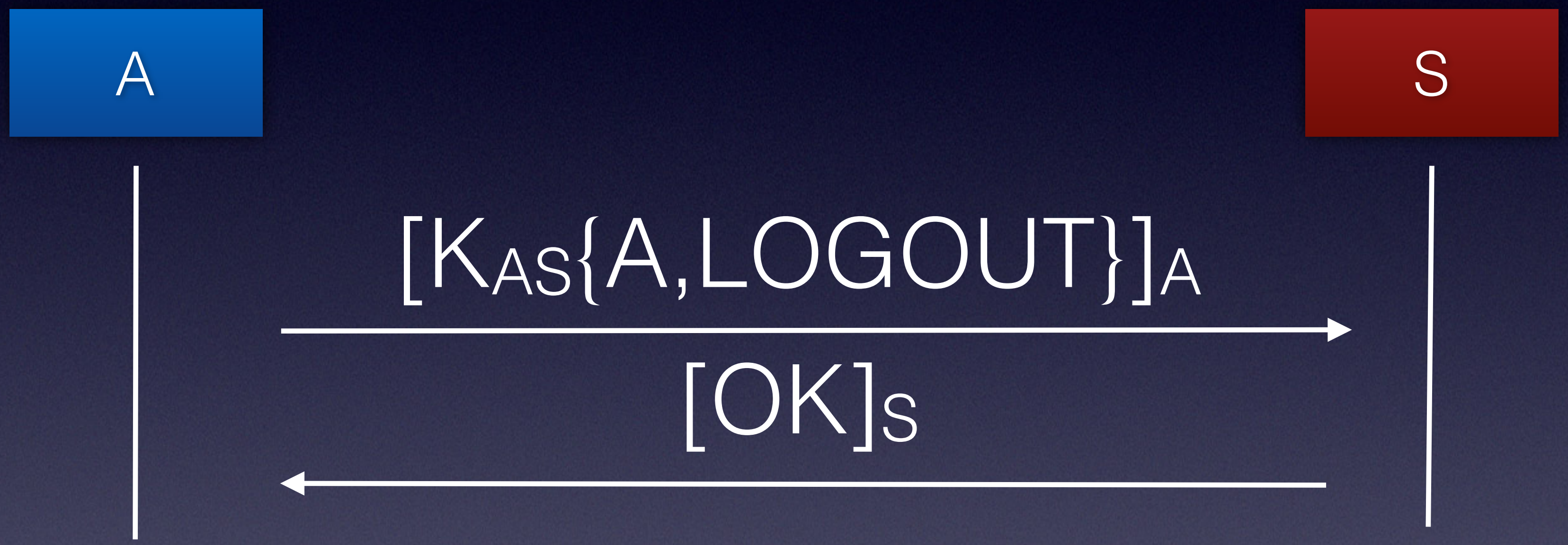
S

$[K_{AS}\{A,*\}]_A$

$[K_{AS}\{(B, P_B, IP:PORT),\ldots\}]_S$

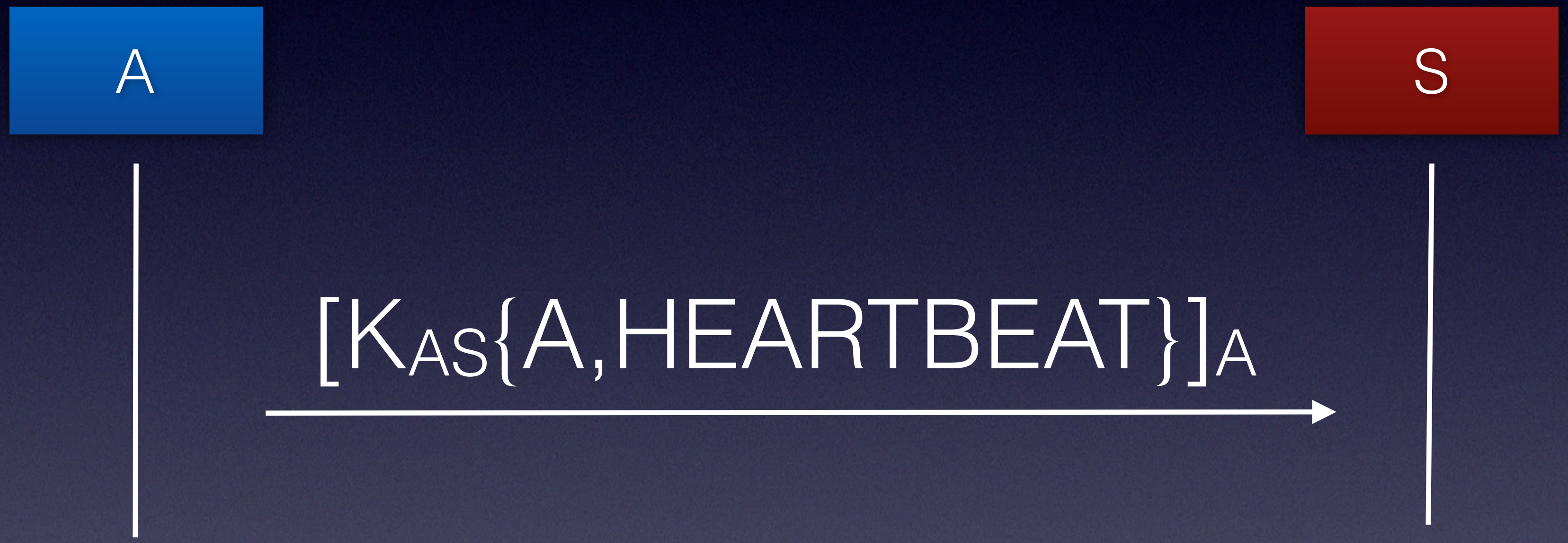
- A can also ask for specific clients details.
- A uses this to get public keys of other clients.

Logout



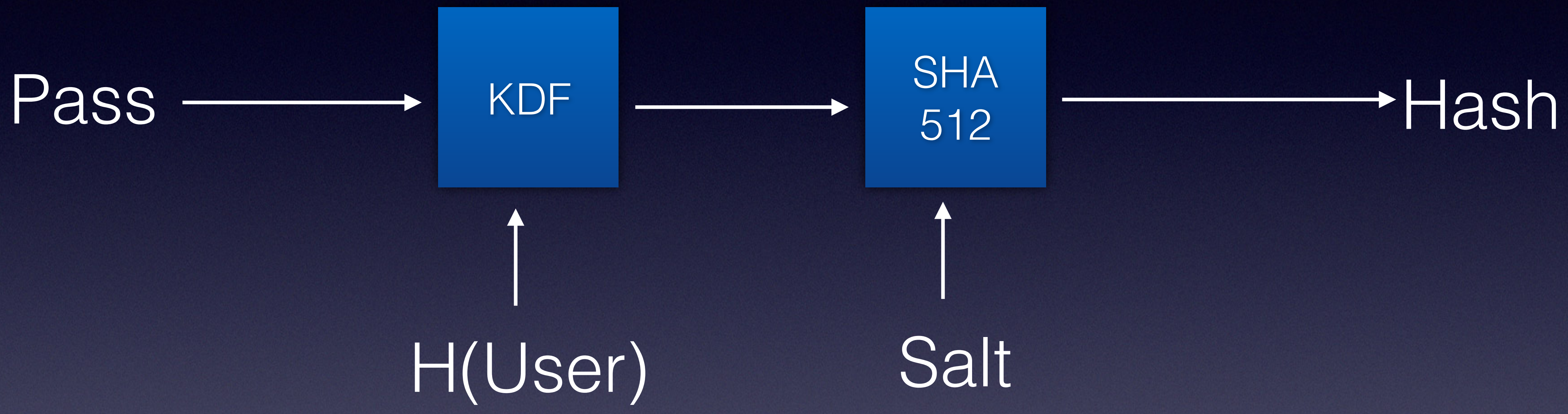
- The Server uses IP:PORT to identify user in cases like this.
- After send an OK, the server broadcasts a message stating that A has logged out
- The Broadcast message looks like $[IP:PORT,LOGOUT]_S$

Heartbeat



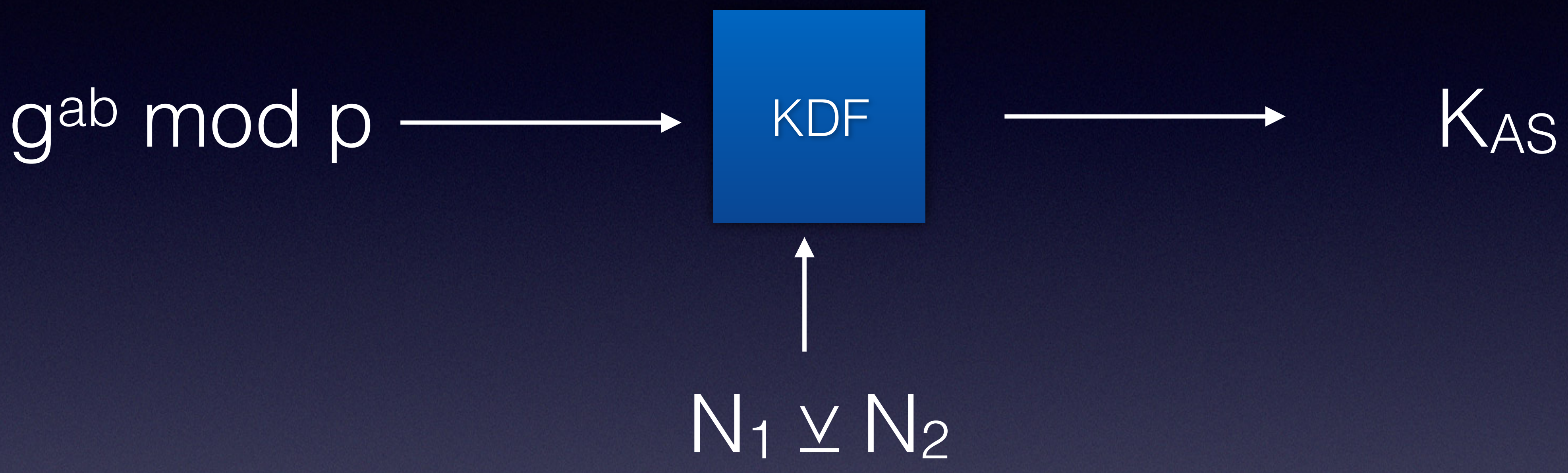
- If the Server doesn't get HeartBeat from Client for 60s then it is logged out and a message is broadcasted indicating A has logged out.

Password Hash Generation



- The Final Hash is stored on Server along with Salt. Server stores this information in a database.
- The Client computes the KDF (H(P)) and sends it to Server where the Server computes the Hash and compares.

Key Generation



$g^{ab} \bmod p$

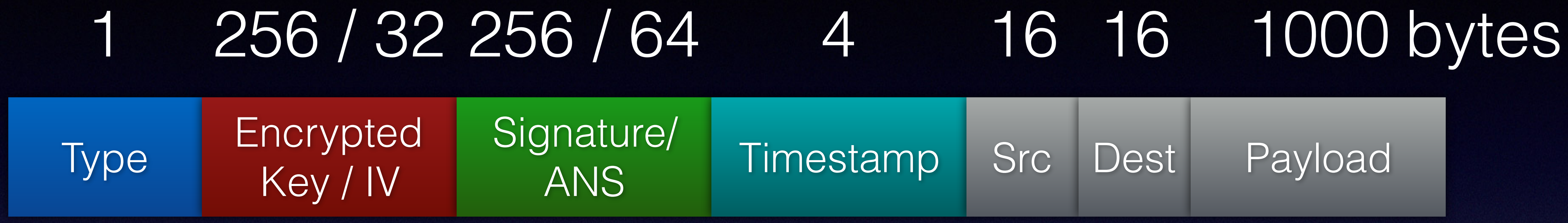
KDF

K_{AS}

$N_1 \veebar N_2$

- The output of DH is given to KDF with $N_1 \veebar N_2$ as salt to get K_{AS}

Packet Format



1	256 / 32	256 / 64	4	16	16	1000 bytes
Type	Encrypted Key / IV	Signature/ ANS	Timestamp	Src	Dest	Payload

- The Entire Grey Region is Encrypted by Key.
- Encrypted Key has K + IV + Tag , if using RSA, else it has IV + Tag only.
- Signature = Sign[timestamp, Grey Region]
- Server or Client Checks Timestamp first after which it verifies the signature. Encrypt then Sign Methodology will be used.
- After the check is passed then the server will decrypt the message.

Stretch Goals

- Dynamically Change the Difficulty of the Puzzles
- Network Reliability like resending a message if it was rejected by server.