2019-2020

# Four in a Row

Cybersecurity Project Documentation

Nicola Barsanti Gianluca Tumminelli

# Contents

# Software Description

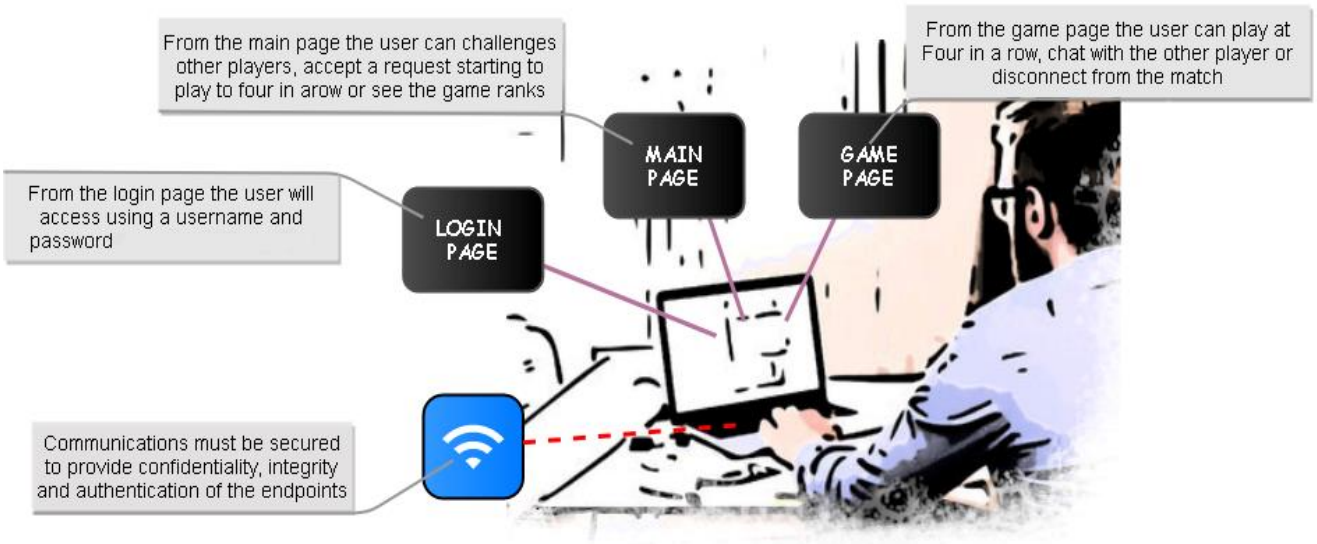
This paper will document the development of the **Four In A Row Online**, an online multiplayer video game used through a prompt interface.

The registered users of the service can access by providing a valid username and password through the ***Login Page***.

Once connected, from the ***Main Page*** they will be able to see all active users, a rank and a list of the challenge requests received from other players. They will also be able to accept one of the received challenges or by choosing a user, send a challenge against him.

The game developed is the classic Four In A Row and it will be played in the ***Game Page***, the game is based on a 6x7 grid in which it is possible in turn to insert a token. The first player who manages to insert 4 tokens in a row wins the game, vice versa if the grid is complete without any winner it ends with a tie.

The confidential information of the users must be protected from be stolen by malicious attackers. Each message authenticity must be proved and the service must be robust to malicious and non-malicious threads.



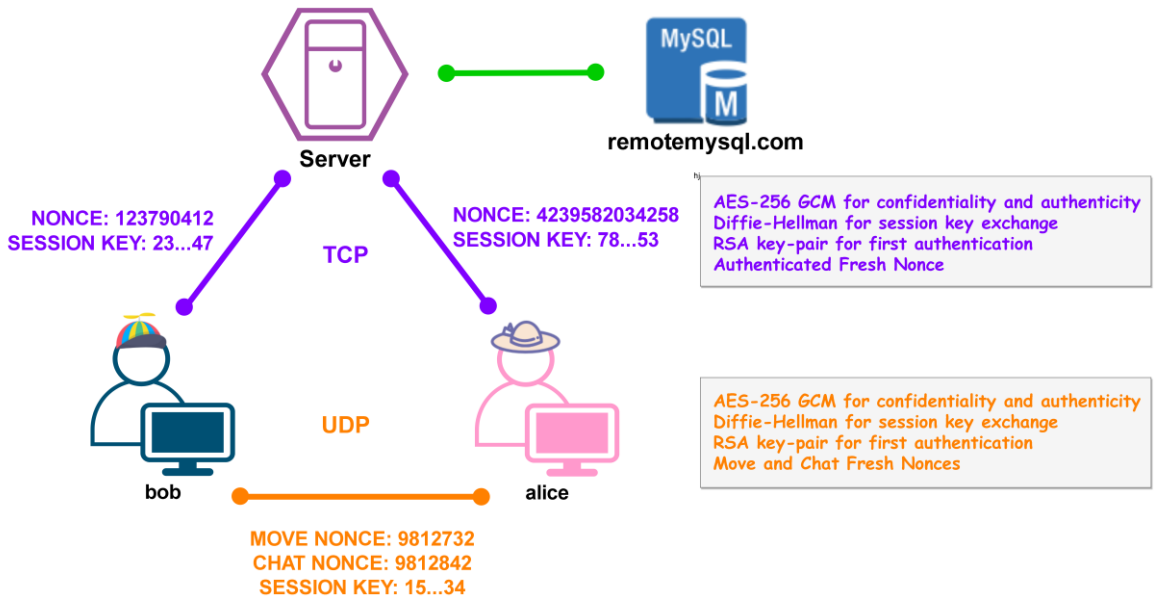
# Service Architecture

## Network Design

The application is delivered by an hybrid communication approach. Each user will have a p2p communication implemented in UDP, to play a match with other players and client-server communication implemented in TCP, to log into the service and perform all the available operations.

## Security Design

Each communication channel will be protected by an AES-256 session key which will be used to encrypt confidential information and authenticate the messages. All the users have also a RSA key-pair to authenticate themselves to the service and other users. To guarantee the freshness of the messages each message will have an authenticated fresh nonce which will be incremented after every completed request to guarantee protection from reply-attack and a signature to guarantee the message authenticity which, depending on the protocol, can be created by the RSA procedure or AES-256 GCM.



# Protocol Architecture

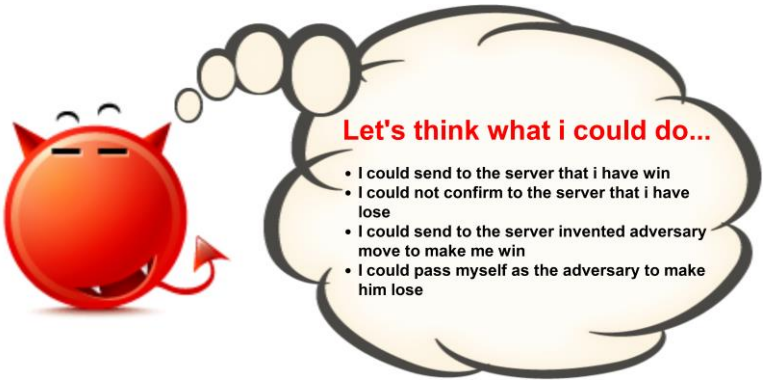
The service is based on a mono-threaded server in which by a shrewd implementation gives the impression of a multi-threaded one. In particular the communication exchanges must be designed to be in a request-response way with no dead-time spent by the server to attempt some responses to complete an interaction(otherwise meanwhile the server is waiting all the users will wait too generating delays in the requests commitment). To perform its operation we have designed 12 possible types of request to the server:

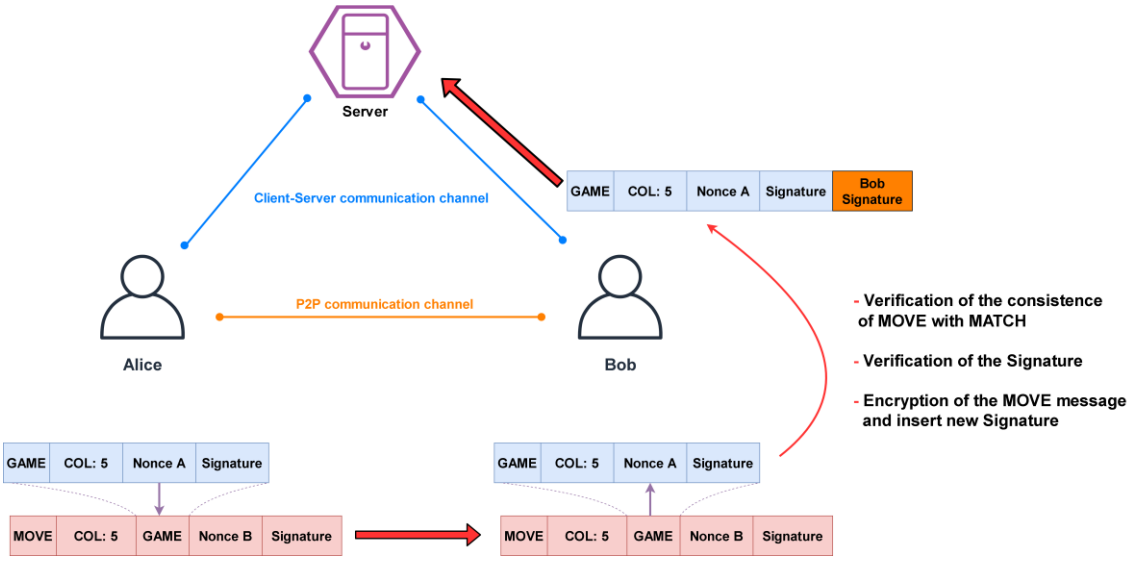
- **CERTIFICATE**: *request the server certificate and perform the initial nonce-exchange*
- **KEY EXCHANGE**: *create an AES-256 key by Diffie-Hellman partials exchange*
- **LOGIN**: *requests access the service by giving an username and password*
- **LOGOUT**: *requests to logout from the service*
- **USER LIST**: *requests a list of all the available users of the service*
- **RANK LIST**: *requests a copy of the game rank from the service*
- **CHALLENGE**: *sends a challenge request to an available user using the server as relay*
- **ACCEPT**: *accepts a received challenge request using the server as relay*
- **REJECT**: *rejects a received challenge request using the server as relay*
- **GAME PARAM:** *exchanges user's parameters needed for client P2P communication*
- **GAME:** *informs the server of the moves given during a match*
- **DISCONNECT**: *leaves the current match informing the server*

And 3 possible types of message that can be exchanged by the users during the playing of a match:

- **KEY EXCHANGE**: *create an AES-256 key by Diffie-Hellman partials exchange*
- **MOVE**: *sends the move of the user during the match*
- **CHAT**: *send a message from the users during a match*

During the game the users use a different channel that hides the information about the progress of the challenge from the server. With this configuration, users could cheat by invalidating the results contained in the ranking. To prevent this we need to provide to the server some form of control and for this reason we have introduced the GAME message. When a player makes a move, he adds a replica signed by his private RSA key to the MOVE message. The receiving player, having checked the consistency of the move made and the one inserted in the attached copy, will sign it and send it to the server which will then be able to verify the progress of the game. The reason for the second signature is to prevent a second form of attack by the cheaters. Our policy is very bad with cheaters, if a player is suspected of having modified a message voluntarily he is punished by automatically making him lose the game. We must therefore prevent a player from impersonating the opponent by voluntarily sending us invalid information to make him lose, this is the reason why a form of authentication is required for both players.



# Protocol Analysis

In the following section there will be described in detail the exchange of the application messages and their structure. We have designed four extra postulates to manage particular situations not covered by the base postulates. During all the analysis with the symbol H we mean an HMAC obtained by some message fields specified with the subscript notation. During the ban analysis we will consider it equivalent to the encryption of all the included fields.

## Certificate Postulate

We need a postulate to link the receive of a server authorized certificate to the obtaining of a valid user public key.

$$\frac{\overset{K_q}{\longmapsto} Q, \{H_{\overset{K_q}{\longmapsto} Q}\}_{K_{ca}^{-1}}, \#(N), \{H_{all}\}_{K_q^{-1}}}{P \mid\equiv \overset{K_q}{\longmapsto} Q, P \mid\equiv Q \mid\equiv \overset{K_q}{\longmapsto} Q}$$

## Signature Postulate

To simplify the BAN Analysis we have made a simple postulate to link the presence of a signature of all the fields of the message to their trustability.

$$\frac{P \mid\equiv \overset{k}{\mapsto} Q, P \triangleleft \{H_X\}_K}{P \mid\equiv Q \mid\sim X}$$

$$\frac{P \mid\equiv P \overset{k}{\leftrightarrow} Q, P \triangleleft \{H_X\}_K}{P \mid\equiv Q \mid\sim X}$$

## Diffie-Hellman Postulates

We need a postulate to link the possession of two Diffie-Hellman parameters to the generation of a shared session key. To simplify the analysis we consider the message and the key the two components to be shared by the two parts independently from what they really are(client-server, client-client)

$$\frac{P \mid\equiv D_1, P \mid\equiv D_2}{P \mid\equiv A \overset{K_{AB}}{\longleftrightarrow} B}$$

We need a postulate to link the freshness of the Diffie-Hellman parameters to the freshness of the shared session key

$$\frac{P \mid\equiv \#(D_1), P \mid\equiv Q \mid\equiv D_2}{P \mid\equiv Q \mid\equiv P \overset{K_{pq}}{\longleftrightarrow} Q}$$

The protocol is used during the client initialization to obtain the server certificate and an authenticated fresh nonce. It is designed to be the first protocol which the clients will execute, as a result of that we have decided to use it also for the sharing of initial nonces necessary to the generation of the shared session key. In this way we prevent randomly chosen nonces susceptible to be a vulnerability for replay attack. The message doesn't require any kind of protection, this is the reason why all the fields are not encrypted and no signature is applied on the request message.


## BAN Logic Analysis

**Real Protocol**

$M_1 \quad C \rightarrow S: \; M, CN$

$M_2 \quad S \rightarrow C: \; M, (C_s, \{H_{C_s}\}_{K_{ca}^{-1}}), CN, SN, \{H_{all}\}_{K_s^{-1}}$

**Ideal Protocol**

$M_1 \quad C \rightarrow S: \; \#(CN)$

$M_2 \quad S \rightarrow C: \; \overset{K_s}{\longmapsto} S, \{H_{\overset{K_s}{\longmapsto} S}\}_{K_{ca}^{-1}}, \#(CN), SN, \{H_{all}\}_{K_s^{-1}}$

**Assumptions**

$C \mid\equiv \#(CN)$

**Goals**

$C \mid\equiv \overset{K_s}{\longmapsto} S$

$C \mid\equiv S \mid\equiv \overset{K_s}{\longmapsto} S$

$C \mid\equiv S \mid\equiv SN$

**Analysis**

M2

$$\frac{C \triangleleft (\overset{K_S}{\longmapsto} S, \{H_{\overset{K_s}{\longmapsto} S}\}_{K_s^{-1}}), \#CN, \{H_{all}\}_{K_s^{-1}}}{C \mid\equiv \overset{K_s}{\longmapsto} S, C \mid\equiv S \mid\equiv \overset{K_s}{\longmapsto} S}$$

*The client has received the server certificate which is validated by the CA. Moreover the message is fresh due to the nonce field and it contains a signature made by the server RSA private key. We can apply the **certificate postulate** to derive that the certificate belongs to the server*

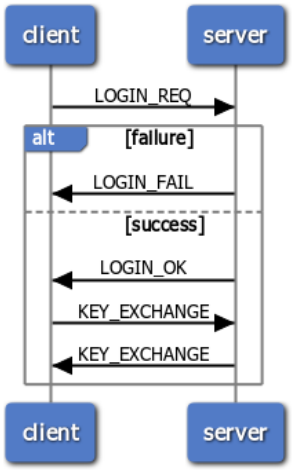
*The client has received a message containing a signature made by all the fields with the server private RSA key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

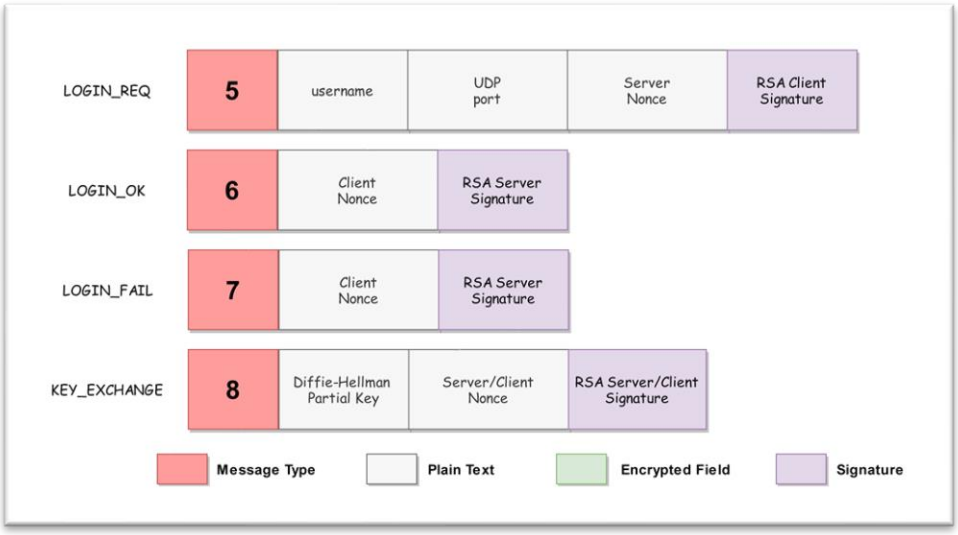
$$\frac{C \mid\equiv \overset{K_s}{\longmapsto} S, C \triangleleft \{H_{all}\}_{K_s^{-1}}}{C \mid\equiv S \mid\sim (SN, CN)}$$

$$\frac{C \mid\equiv \#(CN), C \mid\equiv S \mid\sim (SN, CN)}{C \mid\equiv S \mid\equiv SN}$$

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

The protocol is used by the clients to access to the application. The client has to give a proof of its authenticity by sending a message containing the server nonce obtained from the **Certificate Protocol** and a signature made by its **private RSA key** and the same for the server side. If the server doesn't recognize a user the protocol will end with a **LOGIN_FAIL** message. Otherwise the client and server will proceed with the creation of a session key generated by the **Diffie-Hellman key-generation algorithm**.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C \to S: \; M, U, P, SN, \{H_{all}\}_{K_c^{-1}}$$
$$M_2 \quad S \to C: \; M, CN, \{H_{all}\}_{K_s^{-1}}$$
$$M_3 \quad C \to S: \; M, D_1, SN, \{H_{all}\}_{K_c^{-1}}$$
$$M_4 \quad S \to C: \; M, D_2, CN, \{H_{all}\}_{K_s^{-1}}$$

**Ideal Protocol**

$$M_1 \quad C \to S: \; M, \#(SN), \{H_{all}\}_{K_c^{-1}}$$
$$M_2 \quad S \to C: \; M, \#(CN), \{H_{all}\}_{K_s^{-1}}$$
$$M_3 \quad C \to S: \; M, \#(D_1, SN), \{H_{all}\}_{K_c^{-1}}$$
$$M_4 \quad S \to C: \; M, \#(D_2, CN), \{H_{all}\}_{K_s^{-1}}$$

**Goals**

$$C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S \qquad C \mid\equiv S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} B \qquad S \mid\equiv C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv C \mid\equiv U, P$$

**Assumptions**

$$C \mid\equiv \overset{K_s}{\longmapsto} S \qquad C \mid\equiv S \mid\equiv \overset{K_s}{\longmapsto} S$$
$$S \mid\equiv \overset{K_c}{\longmapsto} C \qquad S \mid\equiv C \mid\equiv \overset{K_c}{\longmapsto} C$$
$$C \mid\equiv \#(D_1, CN) \qquad S \mid\equiv \#(D_2, SN)$$

**Analysis**

M1

$$\frac{S \mid\equiv \overset{K_c}{\longmapsto} C, S \triangleleft \{H_{all}\}_{K_c^{-1}}}{S \mid\equiv C \mid\sim (U, P, \#(SN))}$$

*The server has received a message containing a signature made by all the fields with the client private RSA key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(SN), S \mid\equiv C \mid\sim (U, P, SN)}{S \mid\equiv C \mid\equiv U, P}$$

M2

$$\frac{C \mid\equiv \overset{K_s}{\longmapsto} S, C \triangleleft \{H_{all}\}_{K_s^{-1}}}{C \mid\equiv S \mid\sim (\#(CN))}$$

*The client has received a message containing a signature made by all the fields with the server private RSA key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C \mid\equiv \#(CN), C \mid\equiv S \mid\sim CN}{C \mid\equiv S \mid\equiv CN}$$

M3

$$\frac{S \mid\equiv \overset{K_c}{\longmapsto} C, S \triangleleft \{H_{all}\}_{K_c^{-1}}}{S \mid\equiv C \mid\sim D_1, \#(SN)}$$

*The server has received a message containing a signature made by all the fields with the client private RSA key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(SN), S \mid\equiv C \mid\sim (D_1, \#(SN))}{S \mid\equiv C \mid\equiv D_1}$$

M4

$$\frac{C \mid\equiv \overset{K_s}{\longmapsto} S, C \triangleleft \{H_{all}\}_{K_s^{-1}}}{C \mid\equiv S \mid\sim D_2, \#(CN)}$$

*The client has received a message containing a signature made by all the fields with the server private RSA key. We can apply the **signature postulate** to derive that it believes the message is sent by the server*

*The received message contains a field(nonce), so we can apply the **nonce verification postulate** to derivate that the client believes that only the server could have sent the message*

$$\frac{C \mid\equiv \#(CN), C \mid\equiv S \mid\sim (D_2, \#(CN))}{C \mid\equiv S \mid\equiv D_2}$$

$$\frac{C \mid\equiv D_1, C \mid\equiv S \mid\equiv D_2}{C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S}$$

*We have the two Diffie-Hellman components, we can use the **first Diffie-Hellman postulate** to derive that the client has generate the shared session key*

*We have almost one fresh Diffie-Hellman partial key, we can use the **second Diffie-Hellman postulate** to derive that the shared key is unique and believing to that session*

$$\frac{C \mid\equiv \#(D_1), C \mid\equiv S \mid\equiv D_2}{C \mid\equiv S \mid\equiv (C \overset{K_{cs}}{\longleftrightarrow} S)}$$

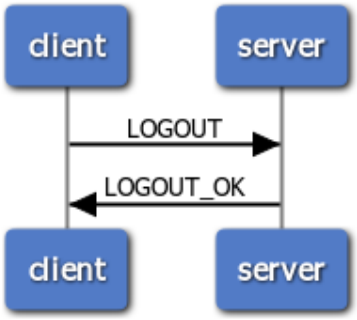
$$\frac{S \mid\equiv D_2, S \mid\equiv C \mid\equiv D_1}{S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S}$$

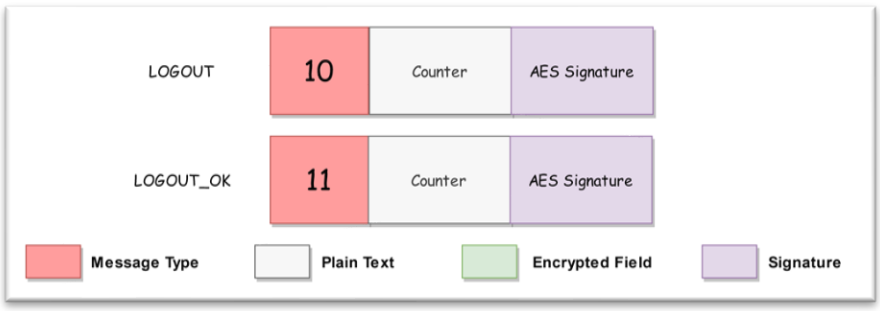
*We have the two Diffie-Hellman components, we can use the **first Diffie-Hellman postulate** to derive that the client has generate the shared session key*

*We have almost one fresh Diffie-Hellman partial key, we can use the **second Diffie-Hellman postulate** to derive that the shared key is unique and believing to that session*

$$\frac{S \mid\equiv \#(D_2), S \mid\equiv C \mid\equiv D_1}{S \mid\equiv C \mid\equiv (C \overset{K_{cs}}{\longleftrightarrow} S)}$$

The protocol will be used by the clients to quit from the application. The messages requires only authenticity and protection to reply attack so they have a signature made by AES-256 GCM based on a counter which will be incremented after each request.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C \rightarrow S: \; M, N, \{H_{all}\}_K cs$$
$$M_2 \quad S \rightarrow C: \; M, N, \{H_{all}\}_K cs$$

**Ideal Protocol**

$$M_1 \quad C \rightarrow S: \; \#(N), \{H_{all}\}_K cs$$
$$M_2 \quad S \rightarrow C: \; \#(N), \{H_{all}\}_K cs$$

**Goals**

$$S \mid\equiv C \mid\equiv N$$
$$C \mid\equiv S \mid\equiv N$$

**Assumptions**

$$C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S \qquad C \mid\equiv S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S \qquad S \mid\equiv C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$C \mid\equiv \#(N) \qquad S \mid\equiv \#(N)$$

**Analysis**

M1

$$\frac{S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, S \triangleleft \{H_{all}\}_{K_{cs}}}{S \mid\equiv C \mid\sim N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C \mid\sim \{N\}}{S \mid\equiv C \mid\equiv N}$$

M2

$$\frac{C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, C \triangleleft \{H_{all}\}_{K_{cs}}}{C \mid\equiv S \mid\sim N}$$

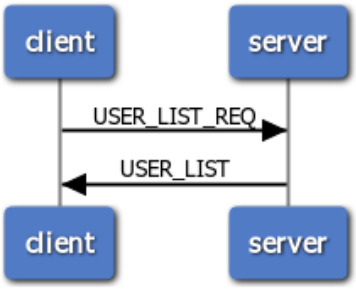
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

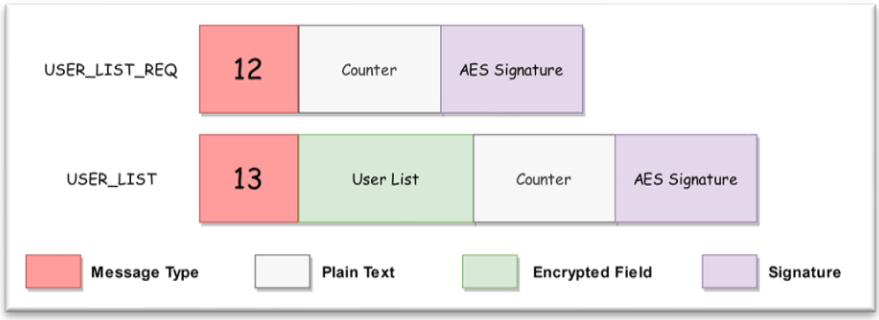
*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believes that only the server could have sent that message*

$$\frac{C \mid\equiv \#(N), C \mid\equiv S \mid\sim N}{C \mid\equiv S \mid\equiv N}$$

## User List Protocol

The protocol will be used from the clients to obtain a list of the users currently available to be challenged. The user list requires to be confidential and it will be encrypted. All the other fields requires only authentication and will be protected by a signature made by AES-256 GCM and based on a counter which will be incremented after each request.




### BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C \rightarrow S: \; M, N, \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \rightarrow C: \; M, N, \{L, H_{all}\}_{K_{cs}}$$

**Ideal Protocol**

$$M_1 \quad C \rightarrow S: \; N, \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \rightarrow C: \; N, \{L, H_{all}\}_{K_{cs}}$$

**Goals**

$$C \mid\equiv S \mid\equiv L$$

**Assumptions**

$$C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$C \mid\equiv \#(N) \qquad S \mid\equiv \#(N)$$

**Analysis**

M1

$$\frac{S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, S \triangleleft \{H_{all}\}_{K_{cs}}}{S \mid\equiv C \mid\sim N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C \mid\sim N}{S \mid\equiv C \mid\equiv N}$$

M2

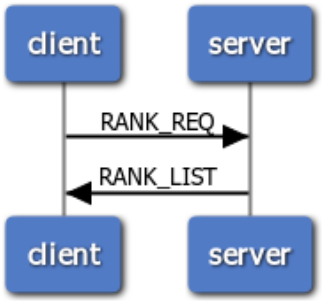
$$\frac{C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, C \triangleleft \{H_{all}\}_{K_{cs}}}{C \mid\equiv S \mid\sim (N, L)}$$

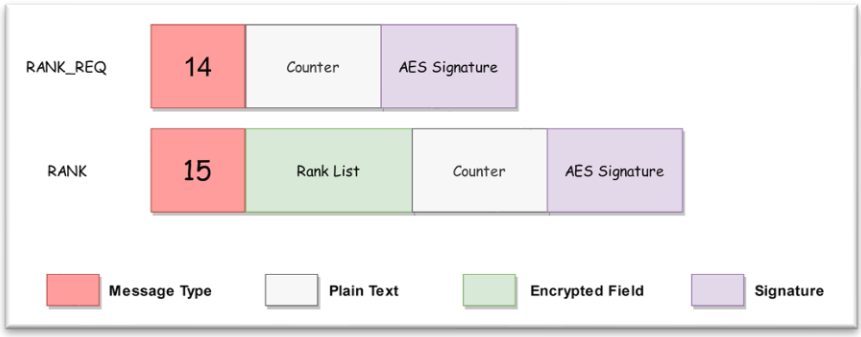
*The client has received a message containing a signature made by all the fields with the AES session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C \mid\equiv \#(N), C \mid\equiv S \mid\sim (N, L)}{C \mid\equiv S \mid\equiv L}$$

The protocol will be used from the clients to obtain a list of the users game statistics. The rank list requires to be confidential and it will be encrypted. All the other fields requires only authentication and they will be protected by a signature made by AES-256 GCM and based on a counter which will be incremented after each request.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C \to S: \; M, N, \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \to C: \; M, N, \{L, H_{all}\}_{K_{cs}}$$

**Ideal Protocol**

$$M_1 \quad C \to S: \; N, \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \to C: \; N, \{L, H_{all}\}_{K_{cs}}$$

**Goals**

$$C \mid\equiv S \mid\equiv L$$

**Assumptions**

$$C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S$$
$$C \mid\equiv \#(N) \qquad S \mid\equiv \#(N)$$

**Analysis**

M1

$$\frac{S \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, S \triangleleft \{H_{all}\}_{K_{cs}}}{S \mid\equiv C \mid\sim N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C \mid\sim N}{S \mid\equiv C \mid\equiv N}$$

M2

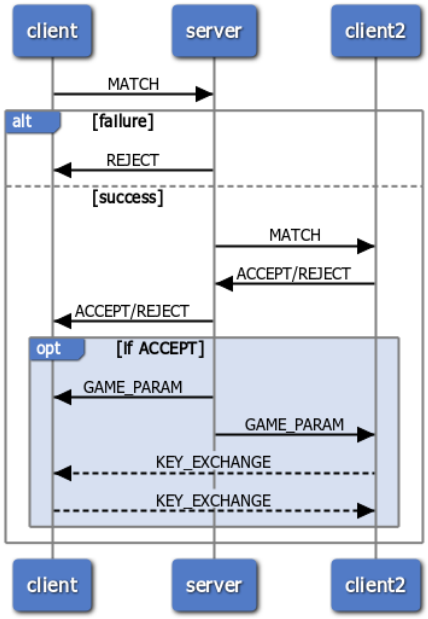
$$\frac{C \mid\equiv C \overset{K_{cs}}{\longleftrightarrow} S, C \triangleleft \{H_{all}\}_{K_{cs}}}{C \mid\equiv S \mid\sim (N, L)}$$

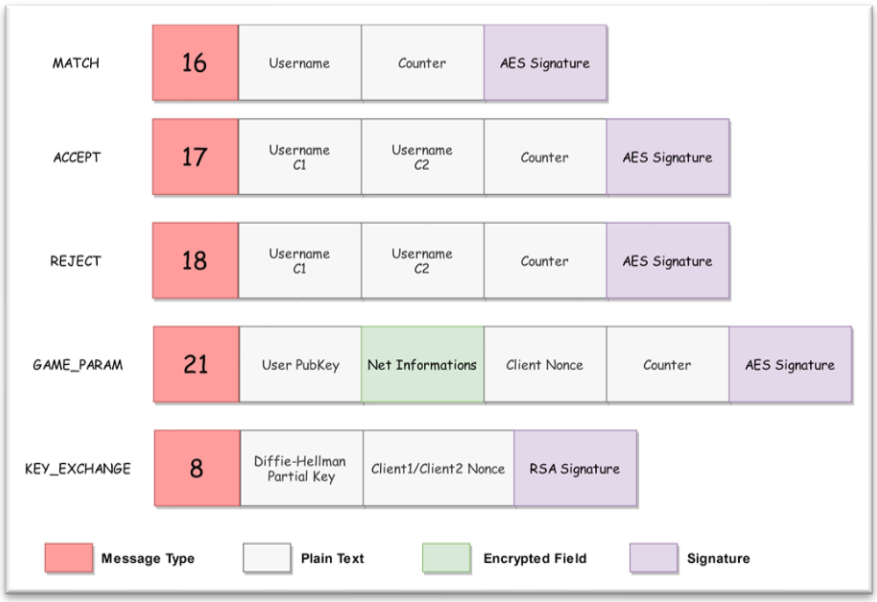
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C \mid\equiv \#(N), C \mid\equiv S \mid\sim (N, L)}{C \mid\equiv S \mid\equiv L}$$

The protocol will be used from the clients to request to another player to join a game. The messages requires authenticity so a signature based on a counter which will incremented after each request and made by AES-256 GCM is applied on each message. The only fields that require confidentiality are the net information of the users and so they will be encrypted.




## BAN Logic Analysis

### Real Protocol

$$
\begin{aligned}
M_1 &\quad C_1 \rightarrow S: \; M, C_2, N, \{H_{all}\}_{K_{sc_1}} \\
M_2 &\quad S \rightarrow C_2: \; M, C_1, N, \{H_{all}\}_{K_{sc_2}} \\
M_3 &\quad C_2 \rightarrow S: \; M, C_1, C_2, N, \{H_{all}\}_{K_{sc_2}} \\
M_4 &\quad S \rightarrow C_1: \; M, C_1, C, 2, N, \{H_{all}\}_{K_{sc_1}} \\
M_5 &\quad S \rightarrow C_1: \; M, K_{C_2}, N_1, N, \{I_{C_2}, H_{all}\}_{K_{sc_1}} \\
M_6 &\quad S \rightarrow C_2: \; M, K_{C_1}, N_1, N, \{I_{C_1}, H_{all}\}_{K_{sc_1}} \\
M_7 &\quad C_2 \rightarrow C_1: \; M, D_2, N_1, \{H_{all}\}_{K_{c_2}^{-1}} \\
M_8 &\quad C_1 \rightarrow C_2: \; M, D_1, N_1, \{H_{all}\}_{K_{c_1}^{-1}}
\end{aligned}
$$

### Goals

$$
\begin{aligned}
&S \mid\equiv C_1 \mid\equiv C_2 \qquad S \mid\equiv C_2 \mid\equiv C_1 \\
&C_1 \mid\equiv S \mid\equiv C_2 \qquad C_1 \mid\equiv S \mid\equiv \overset{C_2}{\mapsto} C_2, I_{C_2} \qquad C_1 \mid\equiv C_2 \mid\equiv C_1 \overset{C_1 C_2}{\longleftrightarrow} C_2 \\
&C_2 \mid\equiv S \mid\equiv C_1 \qquad C_2 \mid\equiv S \mid\equiv \overset{C_1}{\mapsto} C_1, I_{C_1} \qquad C_2 \mid\equiv C_1 \mid\equiv C_1 \overset{C_1 C_2}{\longleftrightarrow} C_2
\end{aligned}
$$

### Assumptions

$$
\begin{aligned}
&C_1 \mid\equiv C_1 \overset{K_{cs_1}}{\longleftrightarrow} S \qquad C_2 \mid\equiv C_2 \overset{K_{sc_2}}{\longleftrightarrow} S \\
&S \mid\equiv C_1 \overset{K_{sc_1}}{\longleftrightarrow} S \qquad S \mid\equiv C_2 \overset{K_{sc_2}}{\longleftrightarrow} S \\
&C_1 \mid\equiv \#(N) \qquad C_2 \mid\equiv \#(N) \qquad S \mid\equiv \#(N) \\
&C_1 \mid\equiv \#(D_1, N_1) \qquad C_2 \mid\equiv \#(D_2, N_1) \qquad S \mid\equiv \#(N_1)
\end{aligned}
$$

### Ideal Protocol

$$
\begin{aligned}
M_1 &\quad C_1 \rightarrow S: \; \#(N), \{H_{all}\}_{K_{sc_1}} \\
M_2 &\quad S \rightarrow C_2: \; \#(N), \{H_{all}\}_{K_{sc_2}} \\
M_3 &\quad C_2 \rightarrow S: \; \#(N), \{H_{all}\}_{K_{sc_2}} \\
M_4 &\quad S \rightarrow C_1: \; \#(N), \{H_{all}\}_{K_{sc_1}} \\
M_5 &\quad S \rightarrow C_1: \overset{C_2}{\mapsto} C_2, \#(N_1, N), \{I_{C_2}, H_{all}\}_{K_{sc_1}} \\
M_6 &\quad S \rightarrow C_2: \overset{C_1}{\mapsto} C_1, \#(N_1, N), \{I_{C_1}, H_{all}\}_{K_{sc_1}} \\
M_7 &\quad C_2 \rightarrow C_1: \; \#(N_1), \{H_{all}\}_{K_{c_2}^{-1}} \\
M_8 &\quad C_1 \rightarrow C_2: \; \#(N_1), \{H_{all}\}_{K_{c_1}^{-1}}
\end{aligned}
$$

**Analysis**

M1

$$\frac{S \mid\equiv C_1 \overset{K_{sc_1}}{\longleftrightarrow} S, S \triangleleft \{H_{all}\}_{K_{sc_1}}}{S \mid\equiv C_1 \mid\sim N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C_1 \mid\sim (N, C_2)}{S \mid\equiv C_1 \mid\equiv C_2}$$

M2

$$\frac{C_2 \mid\equiv C_2 \overset{K_{sc_2}}{\longleftrightarrow} S, C_2 \triangleleft N, \{H_{all}\}_{K_{sc_2}}}{C_2 \mid\equiv S \mid\sim N}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C_2 \mid\equiv \#(N), C_2 \mid\equiv S \mid\sim (N, C_1)}{C_2 \mid\equiv S \mid\equiv C_1}$$

M3

$$\frac{S \mid\equiv C_2 \overset{K_{sc_2}}{\longleftrightarrow} S, C_2 \triangleleft N, \{H_{all}\}_{K_{sc_2}}}{S \mid\equiv C_2 \mid\sim C_1, C_2, N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C_2 \mid\sim C_1, C_2, N}{S \mid\equiv C_2 \mid\equiv C_1, C_2}$$

M4

$$\frac{C_1 \mid\equiv C_1 \overset{K_{sc_1}}{\longleftrightarrow} S, C_1 \triangleleft N, \{H_{all}\}_{K_{sc_1}}}{C_1 \mid\equiv S \mid\sim N}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C_1 \mid\equiv \#(N), C_1 \mid\equiv S \mid\sim C_1, C_2, N}{C_1 \mid\equiv S \mid\equiv C_1, C_2}$$

M5

$$\frac{C_1 \mid\equiv C_1 \overset{K_{sc_1}}{\longleftrightarrow} S, C_1 \triangleleft N, N_1, \{H_{all}\}_{K_{sc_1}}}{C_1 \mid\equiv S \mid\sim N, N_1}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C_1 \mid\equiv \#(N), C_1 \mid\equiv S \mid\sim\overset{c_2}{\longmapsto} C_2, I_{c_2}, N, N_1}{C_1 \mid\equiv S \mid\equiv\overset{c_2}{\longmapsto} C_2, I_{c_2}, N_1}$$

M6

$$\frac{C_2 \mid\equiv C_1 \overset{K_{sc_2}}{\longleftrightarrow} S, C_2 \triangleleft N, N_1, \{H_{all}\}_{K_{sc_2}}}{C_2 \mid\equiv S \mid\sim N, N_1}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

$$\frac{C_2 \mid\equiv \#(N), C_2 \mid\equiv S \mid\sim\overset{c_1}{\longmapsto} C_1, I_{c_1}, N, N_1}{C_2 \mid\equiv S \mid\equiv\overset{c_1}{\longmapsto} C_1, N_1, I_{c_1}}$$

M7

$$\frac{C_1 \mid\equiv \xrightarrow{K_{c_2}} C_2, S \triangleleft \{H_{all}\}_{K_{c_2}^{-1}}}{S \mid\equiv C_2 \mid\sim N_1}$$

*The client has received a message containing a signature made by all the fields with the client RSA private key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

$$\frac{C_1 \mid\equiv \#(N_1), C_1 \mid\equiv C_2 \mid\sim (D_1, N_1)}{C_1 \mid\equiv C_2 \mid\equiv D_2}$$

M8

$$\frac{C_2 \mid\equiv \xrightarrow{K_{c_1}} C_1, C_2 \triangleleft \{H_{all}\}_{K_{c_1}^{-1}}}{C_2 \mid\equiv C_1 \mid\sim N_1}$$

*The client has received a message containing a signature made by all the fields with the client RSA private key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

$$\frac{C_2 \mid\equiv \#(N_1), C_2 \mid\equiv C_1 \mid\sim (D_1, N_1)}{C_2 \mid\equiv C_1 \mid\equiv D_1}$$

$$\frac{C_1 \mid\equiv D_1, C_1 \mid\equiv C_2 \mid\equiv D_2}{C_1 \mid\equiv C_1 \xleftrightarrow{K_{c_1 c_2}} C_2}$$

*We have the two Diffie-Hellman components, we can use the first **Diffie-Hellman postulate** to derive that the client has generate the shared session key*

*We have almost one fresh Diffie-Hellman partial key, we can use the second **Diffie-Hellman postulate** to derive that the shared key is unique and believing to that session*

$$\frac{C_1 \mid\equiv \#(D_1), C_1 \mid\equiv C_2 \mid\equiv D_2}{C_1 \mid\equiv C_2 \mid\equiv (C_1 \xleftrightarrow{K_{c_1 c_2}} C_2)}$$

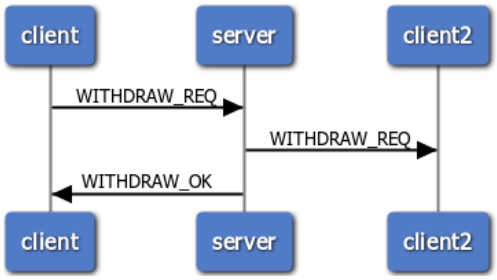
$$\frac{C_2 \mid\equiv D_2, C_2 \mid\equiv C_1 \mid\equiv D_1}{C_2 \mid\equiv C_1 \xleftrightarrow{K_{c_1 c_2}} C_2}$$

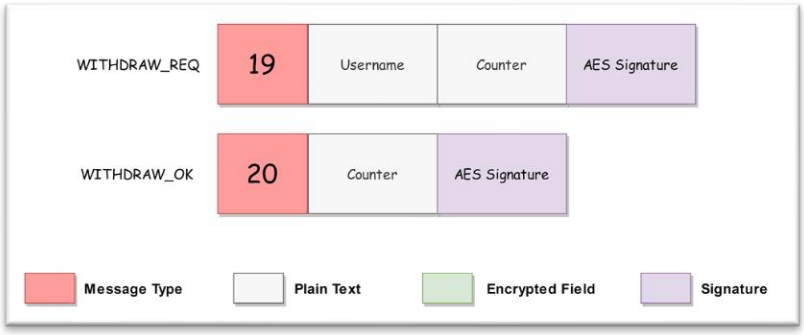
*We have the two Diffie-Hellman components, we can use the first **Diffie-Hellman postulate** to derive that the client has generate the shared session key*

*We have almost one fresh Diffie-Hellman partial key, we can use the second **Diffie-Hellman postulate** to derive that the shared key is unique and believing to that session*

$$\frac{C_2 \mid\equiv \#(D_2), C_2 \mid\equiv C_1 \mid\equiv D_1}{C_1 \mid\equiv C_2 \mid\equiv (C_1 \xleftrightarrow{K_{c_1 c_2}} C_2)}$$

The protocol will be used from the clients to undo a previously sent challenge. The messages requires authenticity so, messages will be protected by a signature made by AES-256 GCM and based on a counter which will be incremented after each request.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C \to S: \ M, U, N, \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \to C: \ M, N, \{H_{all}\}_{K_{cs}}$$

**Ideal Protocol**

$$M_1 \quad C \to S: \ \#(N), \{H_{all}\}_{K_{cs}}$$
$$M_2 \quad S \to C: \ \#(N), \{H_{all}\}_{K_{cs}}$$

**Goals**

$$C \mid\equiv S \mid\equiv N_1$$
$$S \mid\equiv C \mid\equiv U$$

**Assumptions**

$$C \mid\equiv \overset{K_{cs}}{\longleftrightarrow} S$$
$$S \mid\equiv \overset{K_{cs}}{\longleftrightarrow} S$$
$$C \mid\equiv \#(N) \qquad S \mid\equiv \#(N)$$

**Analysis**

M1

$$\frac{S \mid\equiv C \overset{K_{sc}}{\longleftrightarrow} S, S \triangleleft N, \{H_{all}\}_{K_{sc}}}{S \mid\equiv C \mid\sim U}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C \mid\sim U, N}{S \mid\equiv C \mid\equiv U}$$

M2

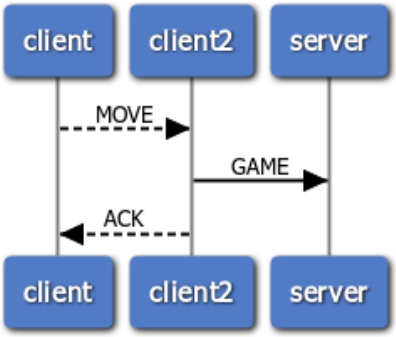
$$\frac{C \mid\equiv C \overset{K_{sc}}{\longleftrightarrow} S, C \triangleleft N, \{H_{all}\}_{K_{sc}}}{C \mid\equiv S \mid\sim N}$$

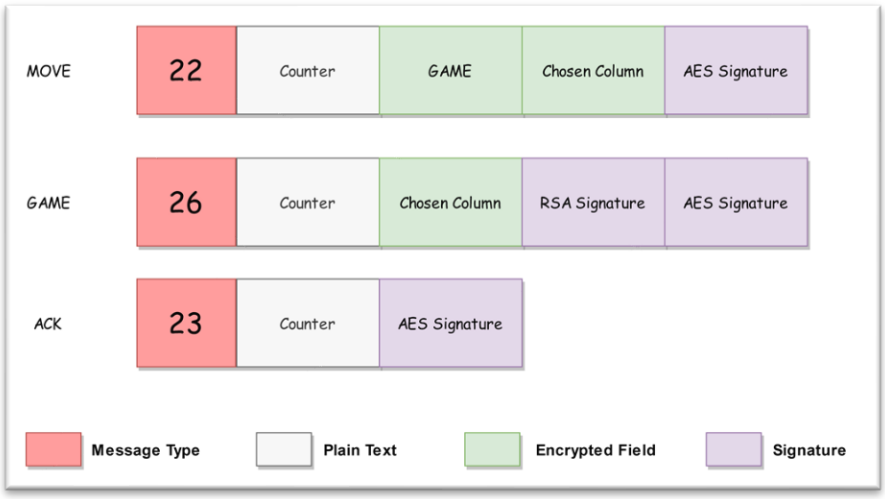
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believes that only the server could have sent that message*

$$\frac{C \mid\equiv \#(N), C \mid\equiv S \mid\sim N}{C \mid\equiv S \mid\equiv N}$$

The protocol will be used from the clients to make a move during the match. The messages requires authenticity so a signature based on a fresh nonce and made by AES-256 GCM is applied on each message. The message contains an entire GAME message encrypted as a field to be passed to the server. More details can be found on the *"The cheater problem"* chapter.




## BAN Logic Analysis

**Real Protocol**

$M_1 \quad C_1 \to C_2 : \; M, CT, \{(C, G, H_{all}\}_{K_{c_1c_2}}$

$M_2 \quad C_2 \to S : \; M, N, \{H_{all}\}_{\overset{K}{\longmapsto} C_1}, \{C, H_{all}\}_{K_{sc_2}}$

$M_3 \quad C_2 \to C_1 : \; M, CT, \{H_{all}\}_{K_{c_1c_2}}$

**Ideal Protocol**

$M_1 \quad C_1 \to C_2 : \; \#(CT), \{(C, G, H_{all}\}_{K_{c_1c_2}}$

$M_2 \quad C_2 \to S : \; M, \#(N), \{H_{all}\}_{K_{c_1}}, \{CT, H_{all}\}_{K_{sc_2}}$

$M_3 \quad C_2 \to C_1 : \; \#(CT), \{H_{all}\}_{K_{c_1c_2}}$

**Assumptions**

$C_1 \mid\equiv C_1 \overset{K_{c_1c_2}}{\longleftrightarrow} C_2)$

$C_2 \mid\equiv C_1 \overset{K_{c_1c_2}}{\longleftrightarrow} C_2)$

$S \mid\equiv C_1 \overset{K_{sc_1}}{\longleftrightarrow} S \quad S \mid\equiv \overset{K_{c_1}}{\longmapsto} C_1$

$S \mid\equiv \#(N) \quad C_1 \mid\equiv \#(CT) \quad C_2 \mid\equiv \#(CT)$

**Goals**

$C_2 \mid\equiv C_1 \mid\equiv C, G$

$C_1 \mid\equiv C_2 \mid\equiv CT$

$S \mid\equiv C_1 \mid\equiv C$

$S \mid\equiv C_2 \mid\sim C, N, H_{C,N}$

**Analysis**

M1

$$\frac{C_2 \mid\equiv C_1 \overset{K_{c_1c_2}}{\longleftrightarrow} C_2, C_2 \triangleleft CT, \{(H_{all}\}_{K_{c_1c_2}}}{C_2 \mid\equiv C_1 \mid\sim CT, C, G}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

$$\frac{C_2 \mid\equiv \#(CT), C_2 \mid\equiv C_1 \mid\sim CT, C, G}{C_2 \mid\equiv C_1 \mid\equiv C, G}$$

M2

$$\frac{S \mid\equiv S \overset{sc_1}{\longleftrightarrow} C_1, S \triangleleft \{H_{all}\}_{K_{sc_1}}}{S \mid\equiv C_1 \mid\sim C, N, \{H_{C,N}\}_{K_{C_1}}}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a signature made by the private RSA key of the client so we can apply the **signature postulate** to derive that the client will believe that the client has sent the fields of the message*

$$\frac{S \mid\equiv \overset{K_{C_1}}{\longmapsto} C_1, S \triangleleft \{\ H_{C,N}\}_{\overset{K_{C_1}}{\longmapsto}}}{S \mid\equiv C_1 \mid\sim C, N}$$

$$\frac{S \mid\equiv \#(N), S \mid\equiv C_1 \mid\sim N, C}{\boxed{S \mid\equiv C_1 \mid\equiv C}}$$

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

M3

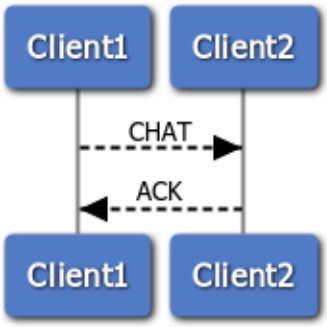
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

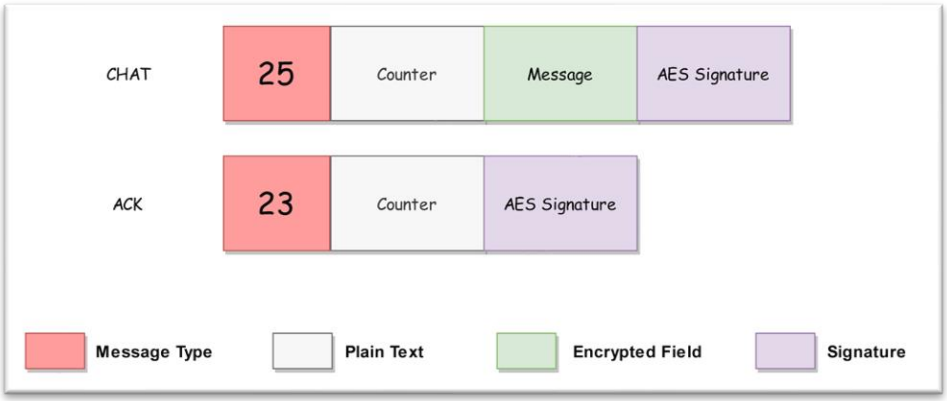
$$\frac{C_1 \mid\equiv C_1 \overset{K_{c_1c_2}}{\longleftrightarrow} C_2, C_1 \triangleleft \{H_{all}\}_{K_{c_1c_2}}}{C_2 \mid\equiv C1 \mid\sim CT}$$

$$\frac{C_1 \mid\equiv \#(CT), C_1 \mid\equiv C_2 \mid\sim CT}{\boxed{C_2 \mid\equiv C_1 \mid\equiv CT}}$$

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

The protocol will be used from the clients to send a message the adversary during the match. The messages requires authenticity so a signature based on a fresh nonce and made by AES-256 GCM is applied on each message. The only field that requires confidentiality is the sent message and so it will be encrypted.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C_1 \to C_2 : M, N, \{ C, H \}_{K_{c_1 c_2}}$$
$$M_2 \quad C_2 \to C_1 : M, N, \{H_{all}\}_{K_{c_1 c_2}}$$

**Ideal Protocol**

$$M_1 \quad C_1 \to C_2 : \#(N_1), \{ C, H \}_{K_{c_1 c_2}}$$
$$M_2 \quad C_2 \to C_1 : \#(N_1), \{H_{all}\}_{K_{c_1 c_2}}$$

**Assumptions**

$$C_1 \mid\equiv C_1 \overset{K_{c_1 c_2}}{\longleftrightarrow} C_2$$
$$C_2 \mid\equiv C_1 \overset{K_{c_1 c_2}}{\longleftrightarrow} C_2$$
$$C_1 \mid\equiv \#(N) \quad C_2 \mid\equiv \#(N)$$

**Goals**

$$C_2 \mid\equiv C_1 \mid\equiv C$$
$$C_1 \mid\equiv C_2 \mid\equiv N_1$$

**Analysis**

M1

$$\frac{C_2 \mid\equiv C_1 \overset{K_{c_1 c_2}}{\longleftrightarrow} C_2, C_2 \triangleleft N, \{(C, H\}_{K_{c_1 c_2}}}{C_2 \mid\equiv C_1 \mid\sim N, C}$$

*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

$$\frac{C_2 \mid\equiv \#(N), C_2 \mid\equiv C_1 \mid\sim N, C}{C_2 \mid\equiv C_1 \mid\equiv C}$$

M2

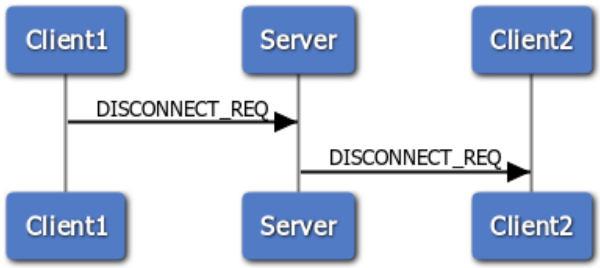
$$\frac{C_1 \mid\equiv C_1 \overset{K_{c_1 c_2}}{\longleftrightarrow} C_2, C_1 \triangleleft N_1, \{H_{all}\}_{K_{c_1 c_2}}}{C_1 \mid\equiv C_2 \mid\sim N}$$

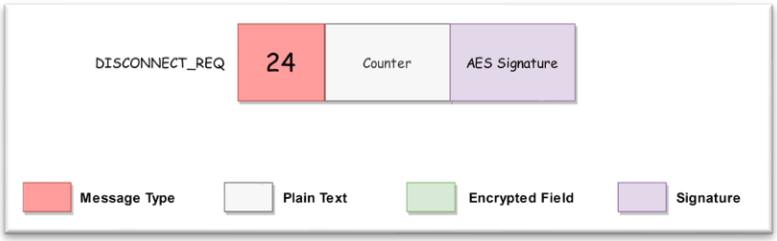
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the other client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the other client could have sent that message*

$$\frac{C_1 \mid\equiv \#(N), C_1 \mid\equiv C_2 \mid\sim N}{C_1 \mid\equiv C_2 \mid\equiv N}$$

The protocol will be used from the clients to exit from a match. The messages requires authenticity so a signature based on a fresh nonce and made by AES-256 GCM is applied on each message.




## BAN Logic Analysis

**Real Protocol**

$$M_1 \quad C_1 \rightarrow S: \; M, N, \{(H_{all}\}_{K_{sc_1}}$$
$$M_2 \quad S \rightarrow C_2: \; M, N, \{H_{all}\}_{K_{c_2 s}}$$

**Ideal Protocol**

$$M_1 \quad C_1 \rightarrow C_2: \; \#(N), \{(H_{all}\}_{K_{sc_1}}$$
$$M_2 \quad C_2 \rightarrow C_1: \; \#(N), \{(H_{all}\}_{K_{c_2 s}}$$

**Assumptions**

$$C_1 \mid\equiv C_1 \overset{K_{c_1 s}}{\longleftrightarrow} S$$
$$C_2 \mid\equiv C_2 \overset{K_{c_2 s}}{\longleftrightarrow} S$$
$$S \mid\equiv C_1 \overset{K_{c_1 s}}{\longleftrightarrow} S$$
$$S \mid\equiv C_2 \overset{K_{c_2 s}}{\longleftrightarrow} S$$
$$S \mid\equiv \#(N) \quad S \mid\equiv \#(N)$$

**Goals**

$$S \mid\equiv C_1 \mid\equiv N_1$$
$$C_2 \mid\equiv S \mid\equiv N_1$$

**Analysis**

M1

$$\frac{S \mid\equiv C_1 \overset{K_{c_1 s}}{\longleftrightarrow} S, S \triangleleft N, \{(H\}_{K_{c_1 s}}}{S \mid\equiv C_1 \mid\sim N}$$

*The server has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the server will believe that the client has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the server will believe that only the client could have sent that message*

$$\frac{S \mid\equiv \#(N), S \mid\equiv C_1 \mid\sim N}{S \mid\equiv C_1 \mid\equiv N}$$

M2

$$\frac{C_2 \mid\equiv C_2 \overset{K_{c_2 s}}{\longleftrightarrow} S, C_2 \triangleleft N, \{H_{all}\}_{K_{c_2 s}}}{C_2 \mid\equiv S \mid\sim N}$$

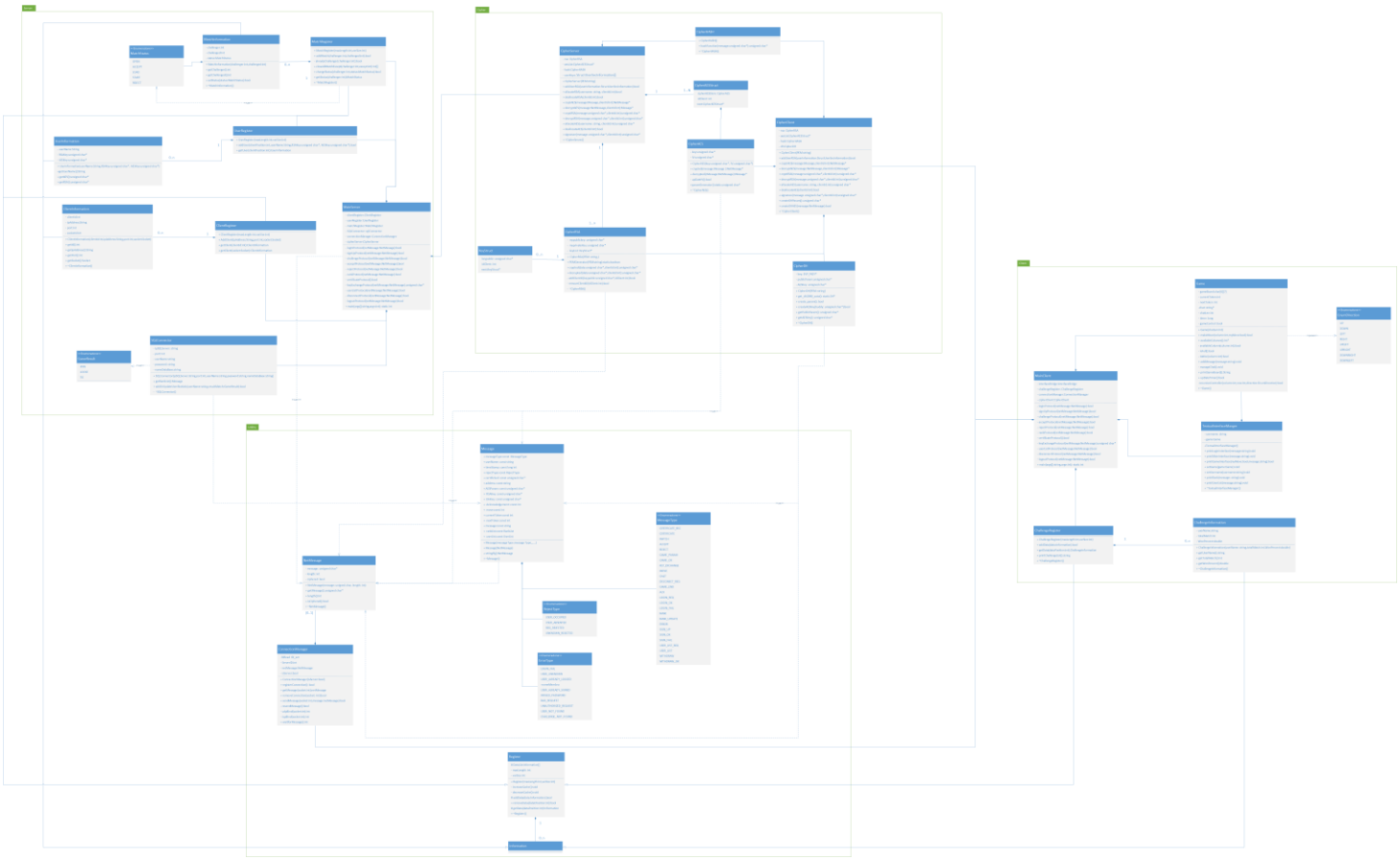
*The client has received a message containing a signature made by all the fields with the AES-256 session key. We can apply the **signature postulate** to derive that the client will believe that the server has sent the fields of the message*

*The message contains a fresh field(nonce), so we can apply the **nonce verification postulate** to derive that the client will believe that only the server could have sent that message*

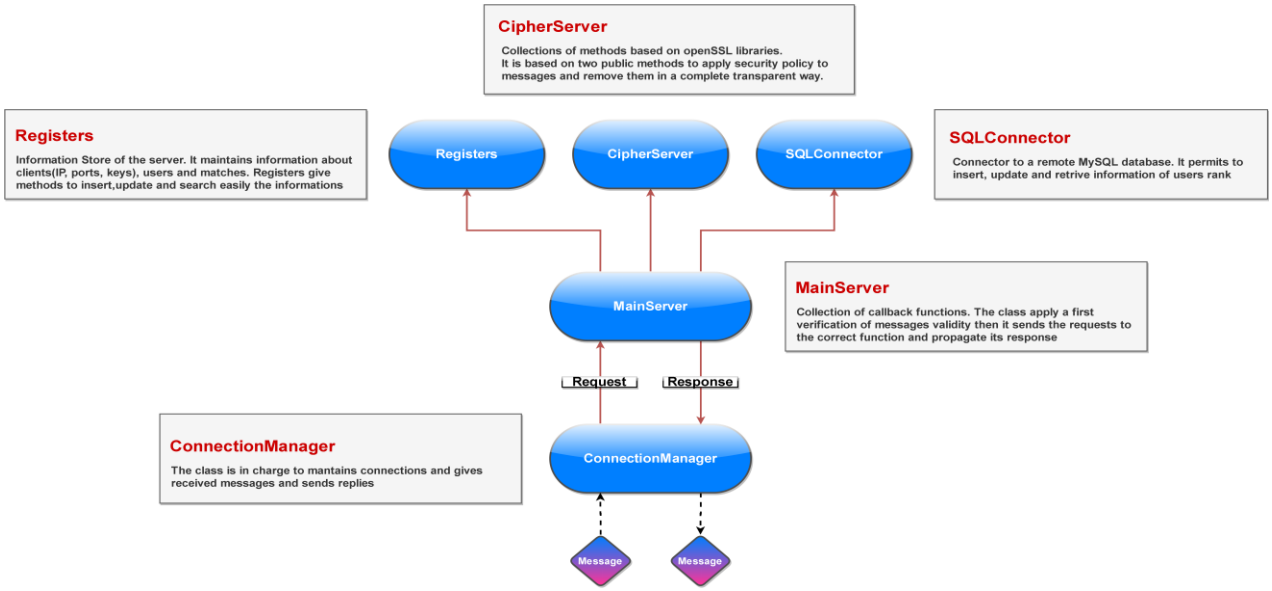
$$\frac{C_2 \mid\equiv \#(N), C_2 \mid\equiv S \mid\sim N}{C_2 \mid\equiv S \mid\equiv N}$$

# Software Architecture

The following section will describe the client and server implementation. The application is created with the C ++ language using OpenSSL and MySQL libraries to create the encryption components and the interface to a remote database granted free of charge by the ***myremotesql.com*** site.



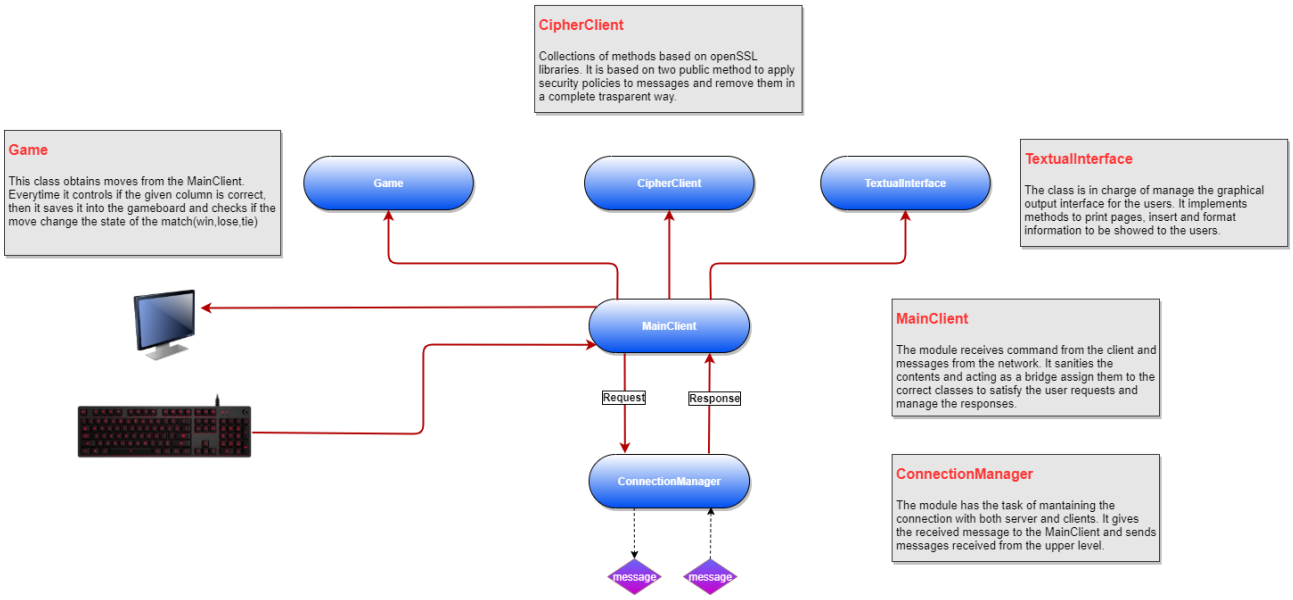
The server is built above **ConnectionManager**, the class is in charge of receiving and sending messages over the network. The behavior of the service is implemented through a set of callback functions developed to manage the requests used by the service(**MainServer** class).




To manage requests, the callback functions rely on three different classes:

- **CipherServer**: the class provides methods to encrypt and decrypt messages by AES-256 GCM, make and verify signatures both by AES-256 GCM and RSA, and generate random values.
- **MysqlConnector**, provides an interface to query a remote database to take or update information about users'ranking
- **Registers**, a set of three classes (**ClientRegister**, **UserRegister** and **MatchRegister**) to manage the information of users registered to the service and their relationships

The client consists of a main module called **MainClient** with the task of executing the commands requested by the user through the keyboard. Also, relying on the **ConnectionManager** it receives messages over the network from the server or, during the execution of a match from other clients through a UDP connection and it is in charge to manage them by a set of callback functions.




To manage requests, the callback functions rely on three different classes:

- **CipherClient**: the class provides methods to encrypt and decrypt messages by AES-256 GCM, make and verify signatures both by AES-256 GCM and RSA, and generate random values.
- **Game**, the class is in charge to manage a match. It receives a column index from the **MainClient** and after have verified that the index is a value between 0 and 6 inserts a token in the respective column. Then it verifies if the game is over and a possible winner
- **Textual Interface**, the class allows the client to display on the screen the answers coming from the server and the moves and messages coming from a possible opponent.

# User Manual

For the application, the server must first be running. First install the mysql connector required to build the server application

```
sudo apt-get -y install libmysqlcppconn-dev
```

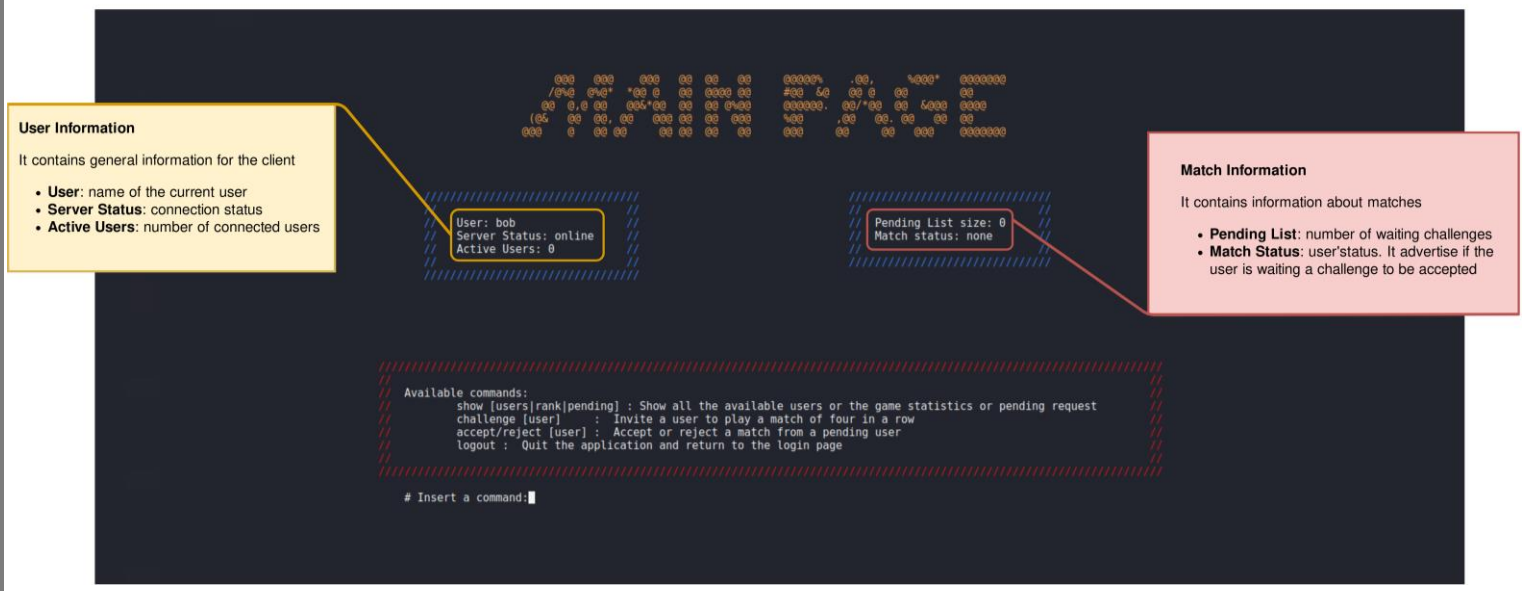
Then you can launch the server application, to do it, move to the **FourInARow** folder and run the bash **launchServer** script. When the server is in execution then you can start a client using the bash **launchClient** script. At the end of the compilation you will be prompted to insert a socket which will be used as a UDP port for playing games.

```
nico@kali:~/Documenti/CyberSecurity$ cd FourInaRow/
nico@kali:~/Documenti/CyberSecurity/FourInaRow$ ls
data  launchClient.sh  launchServer.sh  src
nico@kali:~/Documenti/CyberSecurity/FourInaRow$ ./launchServer.sh
```

The client on first launch will ask you to center the run page. You will only be asked the first time, the following will use the setting entered. To change it, you will need to delete the **screen_size.conf** cache file contained in the **FourInaRow/data** folder.

Apart from the first run the first page that the client will show will be the **login page** where you can access by providing a valid **username** and **password** or close the application.


Once logged in, the main page of the application will be shown. From the page you can challenge other users, see the available ones or those who have challenged you. It is also possible to see the ranking of the players.



Once a match has started by accepting a request received (or your request has been accepted by the challenger) the page for the game will be displayed. It contains a small chat to communicate with the opponent and a representation of the game table. It is possible at any time to cancel a match and return to the main page, but remember that you will be assigned a defeat