CSE 1004 | Network and Communication

Theory DA - 1

Sharadindu Adhikari | 19BCE2105

Well, over the past few years, since its inception, WhatsApp has attracted a lot of attention, because of its friendly user-interfaces, E-2-E encryptions, and, a first of its kind.

The main reason I've chosen to write about WA is due to its mass user-reachability — the billion+ people it connects on a montly basis and the scale at which this real time application operates. From personal usages to small slack replacement, from UPI payments to small scale customer services — WhatsApp has become a one-stop-solution — more like a preferred app over its competitions. And the amount of people it's impacting is huge.

Here, I'm discussing its purpose of reinstating users trust in using chat-apps without worrying about privacy and security concerns, along with various protocols that help govern it.

□ Application Layer Protocol :

From a quick search on the internet I've found that WhatsApp uses a custom XMPP server, built on Erlang to handle the messaging backend.

XMPP (Extensible Messaging and Presence Protocol) is an application layer protocol, mostly like HTTP where the client opens the socket with the XMPP server and keeps it open as long as the client is logged in. It's not like the regular REST API where the

client opens the socket send/receive the data and close the socket. The socket is opened as long as you are signed in. In case of WhatsApp, that's an eternity (not really, WhatsApp reconnects automatically if the connection terminates).

The advantage of using XMPP is, it comes with built in support for rosters and presence, and so as I've mentioned, you don't have to manually setup infrastructure and code to manage subscriptions.

As far as actual technology goes, WhatsApp uses a heavily customized version of Smack library on Android to built their client and uses customized eJabbered Server to handle the XMPP traffic. They might have different backend solution for handling the data though. On iOS and other platforms, I suppose they might have developed their own libraries.

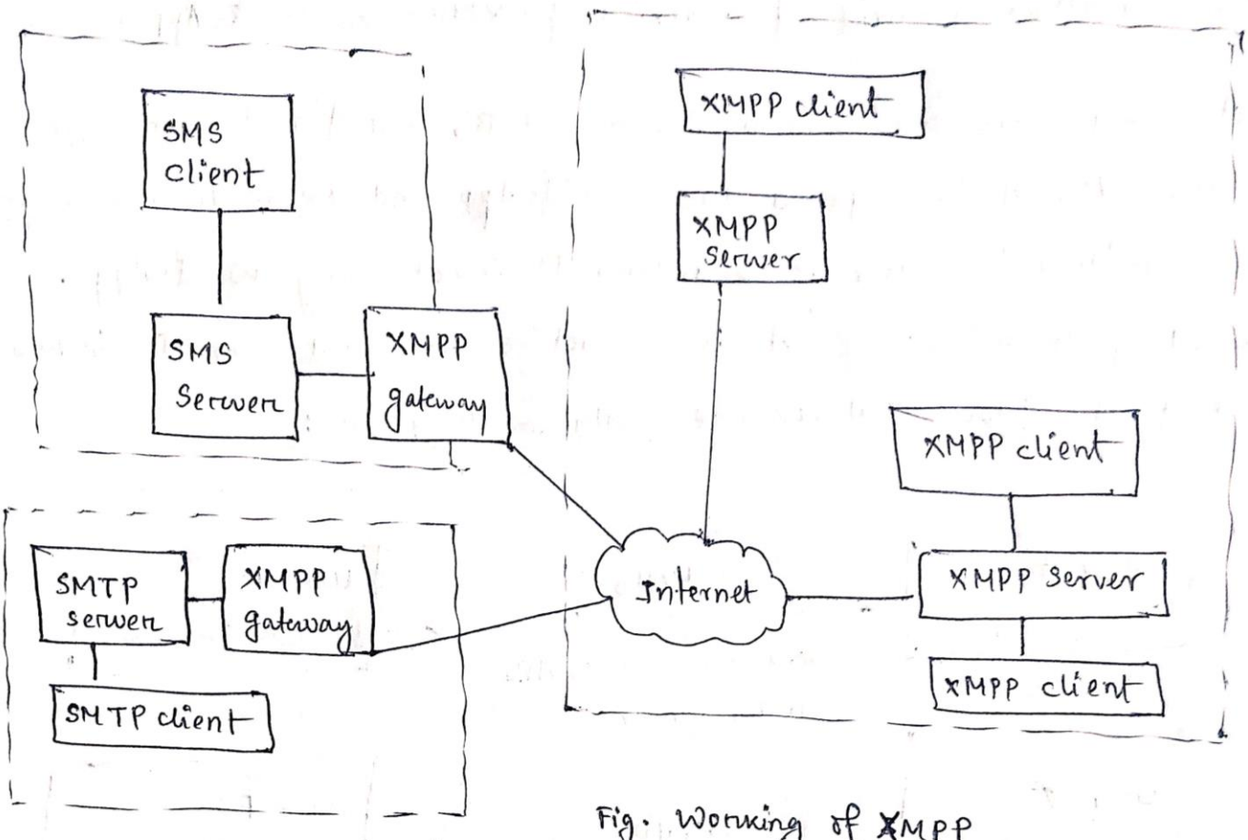


Fig. Working of XMPP

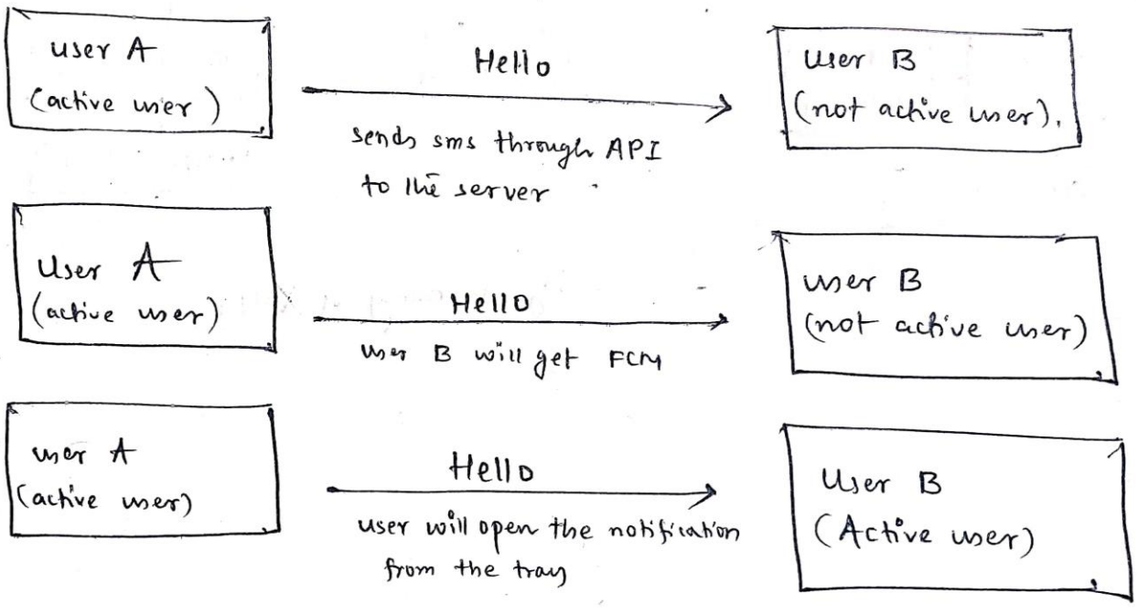
So now let's put together all above information and jot down a scenario about its application.

A user A has internet and a client application (whatsApp) installed on say an Android phone. Another user B has it installed on iPhone. Basically, cross platform.

A types in a message 'Hi' to B. The menage is passed via Webservices to XMPP server. The XMPP serwer has list of all whatsApp users, based on unique phone numbers (everything is encrypted as I'll discuss later on this DA), it will detect B and XMPP serwer will then send menage 'Hi' to B with web service. B will receive this menage on her iPhone. In between, there'll be coding done with buriner rules on server, database level and on client applications as well.

Here's another scenario of working of XMPP on WhatsApp:

Let's say there are two users: A and B, and A sends a menage to B. User A has opened the WhatsApp and he sends a menage, let's say 'Hello' to user B and user B is not using WhatsApp. According to this, User A is an active user and user B is NOT. Rest of the flow I'll convince with a diagram:



Now, according to the above diagram, when user B opens the menage from the notification tray, at that time user B will become an active user and the XMPP will establish the dedicated

connection. Like this, only one-instance of the socket will be live at a time. This also solves the mobile battery draining & internet usages' problems.

☐ From Application Level Security to Privacy & Everything in between:

Many applications, from Web servers to VPNs, rely on secure communication protocols, such as TLSC and IPSec, to protect the RPC communications. Using application level security allows applications to have authenticated remote peer identity, which can be used to implement fine-grained authorization policies.

It may seem unusual for WhatsApp to use a custom security solutions such as ALTS, when majority of internet traffic today is encrypted using TLS. It's because ALTS is designed to be a highly reliable, trusted system that allows for service-to-service authentication and security with minimal user involvement. However, it wasn't always the case.

Only after 4 years of initial launch, did WhatsApp implement the End-to-End encryption policy via Signal protocol. ~~tat~~ WhatsApp calls are encrypted with SRTP, and all client-communications are "layered within a separate encrypted channel".

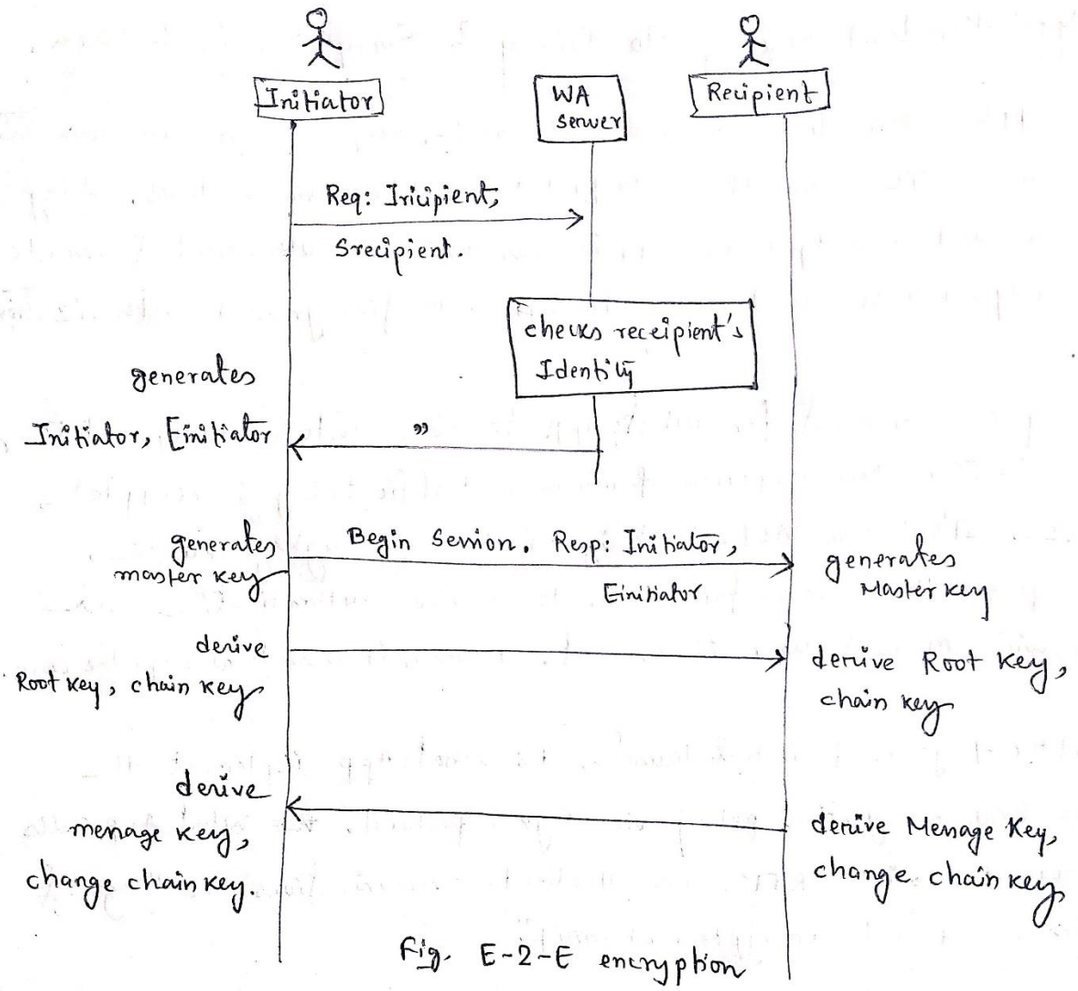


Fig. E-2-E encryption

The Signal Protocol (formerly known as The Text Secure Protocol) is a non-federated cryptographic protocol that can be used to provide E-2-E encryption for voice calls, video calls, & instant messaging convos. This also takes into account the policies of Plausible deniability and Forward Secrecy. Other advantages of using this protocol is its mobile-friendliness of decreasing packet sizes by using protobufs. It works similar to XML but differs by being faster, smaller, and simpler.