Université libre de Bruxelles
INFO-F-405-Project : TORrent

2018-2019

Internet and confidentiality

Confidentiality is known as a major issue over public networks like the internet. Since every packet is routed from a peer to another through a sequence of routers mainly belonging to different entities, additional mechanisms should be put in place in order to ensure data privacy during transmission. The most common way to enable confidentiality (and more...) on the internet is to encrypt HTTP traffic using TLS to get the famous HTTPS protocol. The main idea of this protocol is to protect data using hybrid encryption. When a client reach the server, a (signed) key exchange is performed to share a session key and all subsequent communications between both parties are encrypted using a symmetric (authenticated) encryption scheme.

Nevertheless, even if the content of the communication between the parties is now encrypted, someone watching the network could still learn with whom peers are talking. Indeed, the encryption is only applied on the payload of the packets and not on the headers containing the addresses. The reason is that the routing is obviously impossible if the routers are receiving encrypted addresses.

Onion routing

Onion Routing is a communication procedure aiming at ensuring anonymity over a public network. Anonymity in this context means that two peers can communicate with each others without revealing their true identity and an adversary observing the network would have a really hard time guessing which pairs of user are actually communicating together. The sketch of the routing procedure is the following :

Let $PKE = (E, D)$ be a public-key encryption scheme with encryption function E and decryption function D. Each entity on the network holds a key pair composed of a public key pk and a private keys sk and is reachable at an address IP. We use subscripts to denote ownership of those attributes, e.g. the entity A has public key pk_A, secret key sk_A and is found at IP_a.

When a client C wants to send a message m to a server S, the following happens :

1. The client picks a set of three routers called nodes A, B, C from a public list shared by the whole network.

2. The client encrypts sequentially their message with the public keys of the nodes in order to get a message of the form $ct = E((IP_B, E((IP_C, E((IP_S, E(m, pk_S)), pk_C)), pk_B)), pk_A)$
3. The client sends ct to the entry node A
4. A decrypts the first layer of encryption to get $IP_B, E((IP_C, E((IP_S, E(m, pk_S)), pk_C)), pk_B)$ and forwards the right part to B. Note that A did not learn anything about neither the content of the message nor the identity of the server.
5. B decrypts the second layer of encryption to get $(IP_C, E((IP_S, E(m, pk_S)), pk_C))$ and forwards to the exit node C. Note that it this point, B does not know anything about sender, receiver nor the content of the message. B only routes from A to C by removing a layer of encryption.
6. C decrypts the third layer of encryption to get $(IP_S, E(m, pk_S))$ and forwards it to S.
7. S decrypts and gets the message without learning the address/identity of the client C.

Now that a path between the server and the client is established using three proxy nodes, the connection can resume as a classical HTTP(S) connection. Indeed, each node now knows who is next and previous in the list of routers and can forward accordingly in each directions. In practice, at each step, a symmetric key is exchanged between each participants to avoid all subsequent communications to suffer from the heaviness of asymmetric cryptography. [1]

Goal of the project

In this project, the goal is to create a network in which users can exchange files without learning the identity of the other peers. There exist three types of users :

— The tracker
— Nodes
— Peers

The tracker is unique on the network, its goal is to collect information about the ownership of (parts of) files. Every peer participating to the network will start by a connection to the tracker in order to announce the list of file they can share. Its IP is public and well known by everyone. The nodes are regular onion routing nodes, they do not share files, their only purpose is to relay communications between two peers or between a peer and the tracker. Their IP addresses are also all public and well known by the whole network. The peers are the base users of the system, they want to share files with other peers. They communicate with the tracker to transmit their list of files and to make download request from other peers.

At start, a peer p establishes an onion routed connection to the tracker and sends the list of files it owns. The tracker will answer with a list of all the files available through the network. If p wants one of them, the tracker finds the owner of the file (let us call it p') and links the two exit nodes (the two nodes before him in both chains coming from p and p') together. This will create a tunnel of 6 nodes between p and p'.

1. This high level explanation of onion routing may not be enough to completely grasp the concept from scratch. We strongly advice you to do your own research !

The connection will continue as a normal onion routed connection, the only difference is that we now have 6 nodes between both ends instead of 3. The important property of this system is that at no point the peers revealed their IP addresses to anyone else beside the first node they chose.

We ask you to implement those users in order to run the network on LAN or over the internet. You will provide a code in C++, Java or Python implementing previously described functionalities (tracker, nodes, peers, file exchange) together with a readme explaining how to compile, set up the network and use the software.

One specific task we ask you to achieve is to implement the cryptographic primitives yourself. In a real-life situation, you would rely on external well reviewed libraries, but for educational purpose, you will implement your own crypto. You are free to use any modern cryptographic algorithm you need to solve your problems.

Note that there are a lot of things that were not specified in this document, we only gave an high level overview of what the final product should be. Feel of course free to ask questions but we really want you to make your own decisions regarding the details of your implementation. You can make your own choices, but be ready to explain them !

Deadline

The project should be sent by email to **fragerar@ulb.ac.be** in a single ZIP file with subject "INFO-F-405 Project Group XX" before 23 :55 :00 on Monday, 3rd of December 2018