ECSE 414 Intro. to Telecom. Networks
Deliverable #1 - Description

Camilo Garcia La Rotta #260657037
Jastaj Virdee #260689027
Edward Zhao #260690376

Methodology

The Onion Routing protocol will be implemented with several independent routers that will be referred to as nodes that communicate via TCP and HTTP packets. Each node has a key pair that can be used to implement asymmetric cryptography. Additionally, there will be a special server called the Directory Node. This Directory Node contains a set of all n node address and public key pairs: $\{(a_i, p_i, s_i) | 0 \leq i < n\}$, where a_i, p_i, s_i are the address, public key, and private key corresponding to node i, respectively.

Say a sender $\mathcal{S}$ wishes to send a request (and receive a response) from some destination $\mathcal{D}$. Firstly, $\mathcal{S}$ establishes a TCP connection to the Directory Node, which responds with list of node addresses and public keys to $\mathcal{S}$ describing the nodes that will construct the path that a message will follow.

$\mathcal{S}$ then constructs a request packet destined for $\mathcal{D}$. Given the list of public keys of the nodes in the path, this packet can then be encrypted sequentially. For example, if the path goes from $a_i \rightarrow a_j \rightarrow a_k$, the message is first encrypted with p_i, whose output is encrypted with p_j, whose output is then encrypted with p_k. The encrypted message is then sent to the first intermediary node in the path.

The intermediary nodes all follow the exact same protocol. Say some node N receives a packet from A. Firstly, upon the reception of a packet, the packet body is decrypted with the N's private key. Now, the current node has deciphered enough to send the decrypted packet to its destination, N'. A TCP connection is established between N and N', and the packet is sent. After sending a packet, the node then waits for a response packet. Upon receiving this response, the node may encrypt the response with its *private key*, and send the response back to the node A. At this point, the TCP connection between N and N' may be closed.

When $\mathcal{S}$ receives a response, it will be encrypted with the private keys of all the nodes in the path. However, since $\mathcal{S}$ knows which nodes were in the path, he may decrypt the response with the public keys of those nodes. For this functionality to be guaranteed, the RSA encryption scheme will be used.

The node servers (and Directory Node server) will be programmed with the Python programming language, and will use the Flask framework to handle concurrent requests. The Vagrant tool will be used to automate the creation of virtual machines which will simulate the nodes. Additionally, to save processing power, the Directory Node will run locally on the sender's machine for the purposes of the simulation.

Furthermore, the NTRU cryptoscheme will be considered and investigated to provide key exchange for the TCP connections between nodes that maintains security against quantum opponents.

Testing

The goal of the project is to achieve a minimalist implementation of onion routing and perform analysis of its performance via simulation. As such, testing of this project should be divided into two sections:

1. **Verify and validate the onion routing implementation:** We will follow a Test Driven Development (TDD) paradigm. Writing test cases first, then deriving the unit tests through equivalence partitioning and finally writing the necessary code for all the tests to pass. Throughout

this iterative process we will rely on Continuous Integration (CI) to efficiently manage all the source code generated by each sub-team. The concept of pair programming will be leveraged in order to equally subdivide the tasks among the group and to ensure regular code reviews are performed on the code base. With respect to the application testing, we shall focus on three main aspects of onion routing: bidirectional communication, encryption of data through nodes and randomized path selection. Each of these characteristics will be unit tested with virtual machines and after satisfying our requirements they will be undergo integration testing. In order to efficiently replicate the development and testing environment across the machines of each team member, Vagrant will be used to build and maintain the portable virtual development environments.

2. **Analyze the performance of the application through simulation:** There will be two tools that will be used, one is Nmap which provides a broader overview of the network and Wireshark to examine the packets transferred between nodes. We will be using Nmap for testing enumeration of network components and available application layer protocols at end hosts. It will also be used to identify any vulnerabilities in the network and to ensure the ports are connected to the correct hosts. Wireshark is used to test encryption requirements, provide an analysis of the chosen network and ensure contiguous nodes know the adjacent source destination and not the global ones. For further testing, we may use the mentioned tools such as Brite or Orbis to generate network topology to simulate characteristics in the internet.

BlaBlaBla