CSD Project 2

Kevin Gallagher

Wednesday 15th May, 2024; 17:07

1 Option 2: Communications Privacy

Data privacy is only one aspect of privacy. Every day when people communicate with each other using the Internet they leave traces of metadata that are collected by companies and states. This metadata can inform these entities of who is talking with whom, for how long, and how frequently. Though this may not seem like a big deal at first glance, let's consider the following scenario.

Let's imagine Bob, an average citizen, receives a message from a clinic that specializes in detecting and treating STIs. After receiving this message, Bob quickly sends 5 more messages. These are delivered to Alice, Eve, Christie, Dave, and Frank. Each of these individuals then separately reach out to a doctor or an STI clinic.

As this example illustrates, we were able to tell a lot about the life of Bob, Alice, Eve, Christie, Dave, and Frank without reading the contents of a single message. Simply knowing that a communication existed and when it occurred allowed us to shine a bright light on private aspects of their lives. So, as more and more communications move online it becomes increasingly important that we also protect the privacy not just of the contents of a message, but also the meta-data of that message.

To this end, in this project you will be developing a **threshold mix network**, or a network that protects the meta-data of communications by batching them together and sending them in a random order. This project can be done individually or in groups of two.

To achieve the goals of this project we separate it into three tasks. Each task involves writing and submitting code. Though you may develop on whatever platform, IDE, etc., you wish, **your code must be able to work on a Linux machine.** I do not have any Windows or Mac machines to test the code on, so if the code does not run on a Linux machine your grade will reflect that. To submit your code you will submit a link to your GitHub repository which will contain each of these programs. More information can be found in the submission section below.

1.1 Threshold Mixing - 10 values

In this project you will develop several programs with the aim of protecting communications privacy. Specifically, you will develop a threshold mix with a closed set of participants – that is, the participant list is defined at the creation of the network and does not change. Given this mix, we define a round as every time the threshold mix releases its messages. Participants send messages only when they have messages to send. No clients send chaff traffic, and there is no requirement to send a message every round. More, clients can send and receive more than one message per round.

Your first task in this portion of the assignment is to create a threshold mix server that receives the following when it initializes:

- a static list of clients and their keys.
- a long-term identity key-pair.
- the threshold n

This server will receive encrypted messages from the client, decrypt the header, and send the result to the recipient when the threshold n is reached. Any further messages the mix receives during or after releasing the messages are held for the next round. Remember that the list of participants is closed; your mix must only send and receive messages from the set list of participants. If a message arrives that is not from one of the participants, it must be ignored. After implementing the mix, implement a client that accepts the mix server and its public key at start-up. This client will send messages to and receive messages from this mix. The message format is up to you, but for an extra challenge you can gain inspiration from the Sphinx mix network packet format and protocol.

You may implement this mix in whatever language you see fit, and using whatever libraries you see fit. For public key infrastructure, you can assume that all public keys are somehow automatically verified. The choices of which cryptographic schemes to use are up to you, but I recommend not drifting far away from standard cryptosystems.

1.2 Mix Chain - 5 values

If we trust our mix, the creation of one threshold mix should be enough to hide our metadata from most attacks. However, if we believe that the threshold mix may not be 100% trusted, or if there is a possibility it may become compromised, it is safer to distribute that task among multiple mixes for our clients' protection. Therefore, after implementing the base mix server and client, you must extend this into a simplified linear mix network, which we call a mix chain. So, in addition to the parameters mentioned above please extend the server's functionality to also accept the following at start-up:

- a static list of other mixes and their keys.
- the public keys of the other mixes.

- this mix's position in the mix chain.

In addition to updating its parameters, we also need to update the mix server's functionality. When the mix sends a message, it sends it to the next mix in the network (except for the last mix, which sends it to its recipient). We assume that every mix has a fixed position in the mix network, making it like a chain than a network. However, to simplify our project, this is sufficient.

After extending the mix server to work in a chain, extend the client to accept the list of mixes, their public keys, and their positions, and to encrypt the data multiple times so each mix can remove a layer of encryption. This will ensure that a message cannot be trivially tracked by an adversary who can see messages going into and coming out of the mix. Finally, we have a complete mix chain that can be used to protect clients' communication meta-data.

Please note that not every detail needed to implement the mix chain is provided in this project description. Any other information not included is an implementation decision that should be made by your group. You will all be doing your masters thesis soon, so getting used to doing some research and making these decisions is good practice before you begin your first experience leading a research project.

1.3 Statistical Disclosure - 5 values

It is worth noting that threshold mixes are still possible to attack, One such attack is a **statistical disclosure attack**, or an attack that uses observations from each round to try to determine the probability that client Alice is communicating with client Bob through the mix network. To learn about statistical disclosure attacks, please see the following paper.

http://ndl.ethernet.edu.et/bitstream/123456789/56325/1/8%202005.pdf#page=25

Please download the following CSV.

https://raw.githubusercontent.com/kcg295/CSD-Project-2/main/Anon/Data/mixnet_trace.csv

Each row of this CSV contains the round number, the name of a client of a mix chain, the number of messages that client sent that round, and the number of messages they received that round. If a client sent 0 messages in a round, the client is said to not participate for that round. Using the paper provided above as inspiration, please implement a statistical disclosure attack to determine the different probabilities of Alice's recipients. That is, try to determine with what probability Alice is talking to Bob, with what probability Alice is talking to Christie, with what probability Alice is talking to TheClinic, etc.

There are two possible statistical disclosure attacks in the paper listed above. If you perform the simple attack (Section 2.3 of the paper above), the maximum value you can get for this section is 4 values. If you perform the advanced attack (Section 3.1, Subsection "Complex Senders, Unknown Background Traffic"), you can get up to the full 5 values. For this portion you must submit the code you wrote to perform the statistical disclosure attack, as well as indicate which statistical disclosure attack you opted to attempt in your repository's README. Additionally, you should include the results of your statistical disclosure attack, which will be a list of probabilities, with each index of the list corresponding to another participant in the mix chain.

1.4 Submission

To submit this project, you must submit the link to your repository via email to Kevin Gallagher at k.gallagher@fct.unl.pt before the 7th of June 2024 at 23:59. The subject of the email must contain the string "[CSD Project 2 Option 2 Submission]" and then the student numbers of the members of your group.

You repository must contain at least five programs and a README file. The first program corresponds to the simple mix server. The second program corresponds to the simple mix client. The third program corresponds to the mix server for the mix chain. The fourth program corresponds to the mix client for the mix chain. The final program is to perform the statistical disclosure attack. To reiterate, all of these programs must work on a Linux machine.

The README file of this project must contain the names and student numbers of the members of the group, and must identify the files for each portion of the project and provide instructions on how to compile (if applicable) and run each portion of the project. Your code must be legible and un-obfuscated.

Good luck and have fun!