Chen, Lantian lantian
Espinoza, Dayanna espinoza
Zornberg, Azaria zorn96

# TorText: A Completely Private SMS Platform

## I. Vision

Imagine yourself, for moment, to be a political dissident in a far of nation. You live under an authoritarian government, one that will jail the friends and families of anyone who disagrees with them or challenges them openly. You want to work to change the current regime, and many others share your views. How do you communicate with likeminded individuals though? Meeting in person allows for spies to discover your identities, so you choose to communicate via text messaging from mobile devices. However, this presents a problem; text messaging is not secure, and mobile providers can turn over the content of your conversations to the government. What if we can prevent our messages from being readable by others? Well, while that adds some security, we must remember that this is an unscrupulous, authoritarian government. They can still see whom you talk to, even if they can't see what you say. The moment one person is arrested by the government for being a dissident, they will arrest the rest of you simply because you communicated with a dissident.

Our goal is to create a completely private text messaging, also known as short message service or sms, platform. We believe that for true privacy, we must be able to hide both what you say, as well as who you say it to, and to some extent when you say it. In order to do this, we will have designed a platform that we call TorText.

## II. Background

Two systems that have been developed and are used to attempt at making communication private are onion routing, popularly known as TOR, and End-to-End encryption.

Onion Routing is a technique for anonymous communication through a network. TOR uses multiple machines called "onion routers" to make a socket connection. In order to become anonymous the sequence of onion routers are defined during the connection setup. Each onion router can only know the previous and next hop in the route, so if an eavesdropper gets data from a particular onion router, he/she would not be able to know who was the sender or the receiver. Besides knowing information about the next hop, each router also knows how to get keys for encryption. [1]

Before starting the transmission, the first onion router adds as many layers of encryption as the number of onion routers will be part of the route. Every time the data reaches a new onion router, this router removes the layer and sends it to the next onion router. This allows data to look different at any point in the route, and therefore inhibits eavesdroppers from tracking information. [1]

Although, TOR protects the anonymity of content of data, it does not fully protect the data itself, given that eavesdroppers can probabilistically find out information about conversation partners. If you examine numerous routes originating at one user, it is highly likely that there are few users that exist in a sizable portion of these routes. Therefore, you can determine likely conversation partners by tracking messages for numerous routes and counting the number of occurrences of any given user across routes. Those who occur in many are likely the conversation partners of the original sender. In this way, TOR is insecure, as it does nothing to mitigate this risk.

End-to-End encryption, as discussed in class, is a system of public-key/private-key encryption that allows users to communicate and mask the contents of their messages from any outside observer [2]. The signal protocol, as observed in class, can do this perfectly, in that no outside observer can read messages not intended for them. However, end-to-end encryption does nothing to protect the metadata of a person's conversations. An outside observer can still determine who people are conversing with and when they are conversing.

## III. Component Overview

In order to create a completely private sms platform, we need to accomplish three principal tasks. Firstly, we need to be able to ensure reliability in the platform. Even if TorText can guarantee privacy, it will be considered a failure if it does not meet the basic needs of a communications system. Reliability is one such need; no communication system can be considered usable if its users are unable to determine that their messages have gone through, or to be notified if they cannot successfully be sent. Secondly, we need to be able to guarantee a reasonable speed for communications. We define speed to be the latency between sending a message and receiving confirmation that it was delivered, or knowledge that there was a failure. Because sms was created as a system for sending short messages very quickly, if our system is not sufficiently fast as to not inhibit conversation, then people will choose a less secure system. Finally, we need to be able allow for completely private conversation without compromising either of the two aforementioned aspects of the system.

In order to accomplish these goals, we have introduced two major changes to the Onion Routing protocol and added one new module to the protocol. The two changes are the introduction of what we call Courier Families for choosing routing paths, and the introduction of a system of acknowledgments for the purpose of reliability. The new module we have created shall be known as the Ghost Message module, and is used to allow for privacy of communication despite the aforementioned weaknesses in the Onion Routing Protocol.

## III-A. Assumptions

TorText is founded on two major assumptions. The first is that we can have at least one completely secure, incorruptable server. This server will be used for holding records of public keys and phone numbers, as well as the various mappings of Courier Families to phone number-public key pairs.

The second assumption is that we have a hash function that maps phone numbers to courier families in a way that is uniformly random at both global and local scales. The scale is important to guarantee that there is no case where all phone numbers with one particular area code map to the same courier family. The reason this is necessary to guarantee privacy shall be analyzed later.

## III-B. Courier Families

The onion routing protocol involves choosing a path of intermediaries, the onion's "layers", to carry our message from sender to recipient. However, the method by which these intermediaries are chosen is not strictly defined. For TorText, we have defined a system for choosing intermediaries, or couriers, to carry messages. As mentioned before, every phone number maps to a courier family. At the beginning of each day, each user requests, from the central server, the courier families for every person with whom they intend to converse. The central server then sends the user the phone number-public key pairs for every member of that courier family. The user also requests his or her own courier family. Each courier family shall be comprised of approximately one thousand phone number-public key pairs.

Message chains, for the full route from sender to receiver and for acknowledgments to return to sender, are comprised of one hundred users, roughly 10% of the courier family. The position of the intended recipient in this chain is uniformly, randomly selected. For the number of intermediaries that are needed to carry the message to its intended recipient, the sender selects that number of phone number-public key pairs from the courier family of the recipient. The number of intermediaries that are used to carry the message to the recipient is then appended to the end of the message. The sender will then randomly order the intermediaries and perform the proper steps to encrypt and prepare the message for transmission. The message is finally sent and carried to its intended recipient using the standard onion routing protocol. Because courier families do not change, the chains used to carry messages to particular conversation partners will look similar across different messages. This property becomes important for masking the intended recipient of our messages in conjunction with ghost messages.

## III-C. Acknowledgments

In order to allow users to be notified regarding the success or failure of sending their messages, we have introduced acknowledgments into onion routing. Under Onion Routing, whenever a user receives a message, they decrypt its body with their private key in order to determine to whom they should forward it next. If the message is intended for them, then when the user decrypts the body, it becomes an actual message rather than another encrypted body and forwarding instructions. Under TorText, when this happens, the recipient of the message will note the original sender and send an acknowledgment to them. The acknowledgment itself is just a message with a unique code saying "I got your message with id *i*" where *i* is the serial id that is normally contained in sms messages used for ordering of received messages.

The acknowledgment will be sent back to the message's sender using the same onion routing protocol that was used to send it there. The number of intermediaries to be used to carry the acknowledgment can be calculated as one hundred, less the number of intermediaries used to carry the original message to the recipient. Because the number of intermediaries used to carry the original message to the recipient was appended to the end of the message, this is simple to do. The specific intermediaries for routing the acknowledgment will be chosen randomly from the courier family of the user sending the acknowledgment. This is slightly different from the aforementioned protocol, where the courier family of the user receiving the message is used. The reason for this is so that, to an outside observer, the courier family used in a message's route never changes.

In the case where an acknowledgment is not received for a message, there are three possibilities for what could have occurred: someone in the path may have dropped the original message; someone in the return path may have dropped the acknowledgment; the intended recipient of the message is not reachable on the network. In the case where an acknowledgment is not received within a reasonable timeout window, the original sender will retransmit the message along the original path. If the original message was dropped, or if the acknowledgment was dropped, it may have been a one-time occurrence, and so retransmitting with the original path can be successful. However, if someone in the original path, other than the intended recipient, is not currently reachable on the network, these retransmissions will fail. In this case, after some number of failed retries, a new path will be chosen from the courier family and the original message will be resent on that path. If the sending continues to fail, and a sizable number of different paths also fail, then the likely scenario is that the intended recipient is not reachable on the network. In this case, the sender will be notified that the message cannot be sent at the current time. The specific parameters for the number of retries before choosing a new path, and the number of paths chosen before declaring failure, will have to be determined experimentally through a large scale psychological study to determine values that maximize user happiness with the system.

## III-D. Ghost Messages

One might think, courier families do not guarantee privacy of communication because I can, over a long period of time or over numerous messages, observe all message paths for messages sent by one user; then, with high likelihood, there will only be one user present in all paths, and that user is the person with whom the user is conversing. This observation is correct. Even though, to an outside observer, all message paths look similar, messages never appear stop in one place, and the courier family in a message's path never changes, one could still probabilistically determine a user's conversation partner with high likelihood given sufficient message paths. In order to counteract this, we have created the ghost messaging module.

In order to mask our true conversation partners, we will send out fake messages to the courier families of our conversation partners. These fake messages will simply be garbage text that is encrypted to be sent and carried by one hundred intermediaries before returning back its original sender. Due to the properties of onion routing, the message itself will be indiscernible from a real message as it is sent between any two intermediaries. Because of our addition of acknowledgments, and our random placement of the intended recipient in the path, the paths used for garbage messages will also appear the same as the paths used for real messages. The same retransmission protocol will be used for our ghost messages as for our real ones. We also send enough ghost messages to each courier family such that the expected number of message paths that contain our conversation partners is equal to the expected number of message paths in which any random member of the courier family will appear. If those two values are equal, then our conversation partner is probabilistically indistinguishable from a random member of the courier family.

One might think, "even if I can't determine who within a family you are conversing with, I've still narrowed down your conversation partner to a list of one thousand people." This observation would also be correct. In order to further obfuscate our conversation partners, each user will also send ghost messages to two courier families in which he or she has no conversation partners. Roughly 10% of all messages sent each day will be directed toward these two families, with the division of messages between them being random. If a given user has a sizable number of conversation partners, then the portion of their traffic that flows to a courier family in which they have a conversation partner will look extraordinarily similar to the portion of their traffic that flows to courier families in which they do not. This further inhibits the ability of an outside observer to determine a user's conversation partner.

One might now think, "well, ghost messages might do a bit to mask your conversation partner, and make it so that I cannot probabilistically determine who they are, but humans text in specific patterns, so you can probably filter out ghost messages by pattern." This observation would once again be correct. In order to account for this, we have added two components to the ghost message module: simultaneous layering and time delayed mimic.

Simultaneous layering is the idea that, when you send out a text message, a number of ghost messages are prepared and sent out simultaneously. This causes many ghost messages to perfectly model your texting patterns and ensure that no real messages are ever sent at a time when no ghost messages are sent. Time delayed mimic is a process wherein the system observes the messaging patterns of a user within half hour blocks. These patterns are then copied, and many of the ghost messages are sent out mimicking these messaging patterns randomly distributed over the next day. These blocks of ghost messages mimicking a user may also overlap with other real user messaging. We believe that time delayed mimic may also help, to some extent, to mask when a user is conversing. Generally, we believe that these two components of ghost messaging will make it far more difficult to filter out ghost messages using any form of pattern analysis.

## IV. Analysis

In order to analyze our system, we will examine how the various parameters we have outlined affect TorText's measures of reliability, speed, and privacy.

### IV-A. Reliability

Due to the implementation of acknowledgments, we believe that TorText can be said to be reliable. Given our system of acknowledgments, and the general properties of onion routing and public/private key encryption, a user will always know whether or not his or her message has been delivered. Acknowledgments cannot be forged, so there exists no chance of a user receiving a fake acknowledgment. The system of acknowledgments also assists with masking our conversation partner by causing all message paths to be closed loops.

## IV-B. Latency

We mentioned when we outlined courier families that every message path would have one hundred nodes in it. This number was chosen because it allows us to guarantee a reasonably low latency, while one hundred nodes is sufficiently many such that you cannot easily narrow down a list of conversation partners simply by looking at the first ten messages a user sends.

According to tier 1 carriers within the United States, the average latency for a single sms transmission between any two phones on the tier 1 network is under 200ms [3]. According to studies conducted of nationwide telephone network performance, the nationwide failure rate for single sms transmissions between any two phones is 5% [4]. There are several reasons for these failures, with each one requiring a different number of retries on average to succeed. The three most prominent reasons for failures are sms delivery being postponed by the network, the temporary routing destination for a phone number being changed, and network resources being unavailable.

When the network postpones sms delivery, it takes, on average, 2.5 retries in order to successfully transmit a message. These types of failures make up 86.5% of all failures [4]. When the temporary routing destination for a phone number has been changed, it takes, on average, 1.1 retries in order to successfully transmit a message. These types of failures make up 12.8% of all failures [4]. When a network resource is unavailable, it takes, on average, 2.6 retries in order to successfully transmit a message. These types of failures make up .4% of all failures [4].

Given this, the expected number of transmissions to send a message successfully between any two phones on a tier 1 network can be calculated as 1 + E(retries given failure)*P(failure) = 1 + 2.4075*.05 ~ 1.12. If we expect 1.12 transmissions to send a number between any two phones, and expect the time for each transmission to be under 200ms, then we can expect the time to pass a message along the 100 phones in a TorText route to be (100 hops)*(1.12 transmissions/hop)*(.2s / transmission) ~ 22 seconds. Since the recipient is randomly placed in the route, their expected position is halfway through.

Therefore, the expected latency for a full route is ~22 seconds, which is the time to send a message and receive acknowledgment. The expected time it will take for the intended recipient to get a message is ~11 seconds. We believe that this is sufficiently quick as to not interrupt the flow of conversation.

## IV-C. Privacy

In order to implement the ghost messaging module, we significantly increase the load on the network. Previously, we defined each courier family to have approximately one thousand members, with 10% of them existing in any random message route; this was in order to mitigate the load. In order to make random users probabilistically indistinguishable from conversation partners, we send 9 times as many ghost messages as we do real messages. This was determined as follows:

- Let R be the number of message paths a random member of the courier family is in per day
- Let P be the number of message paths our conversation partner is in per day
- We want E(P) = E(R) to make them probabilistically indistinguishable
- Let m be the number of messages sent to our conversation partner per day
  - Without ghost messages:
  - E(P) = m
  - E(R) = m/10
- Let n be the number of total messages sent including ghost messages
  - E(P) = m
  - E(R) = n/10
- E(P) = E(R)
  - m = n/10
  - n = 10*m
- Number of ghost messages = n-m = 10m – m = 9m
- Generally speaking, for any proportion p of courier family present in a path, the ratio of ghost messages to real message is (1-p)/p

Given this, 1/10th of our messages sent to any courier family with a conversation partner in it will be real messages. We also send 10% of all of our traffic to groups with no conversation partner in it. Therefore, we can calculate our real network utilization as 9%, since it is 10% within the 90% of courier family traffic where we are speaking to someone, and 0% within the 10% of courier family traffic directed to families where we are speaking to nobody. This utilization is directly related to the proportion of each courier family that appears within any given message route. That is why we selected to have courier families contain approximately 1000 members. We also believe that time delayed mimic will somewhat mitigate the strain put on the network by ghost messages; however, it is also possible that our simultaneous layering system can cause some flash crowding of the network, since a portion of the ghost messages are sent at the same time as real messages. Overall, however, we believe that 9% real network utilization is a fair tradeoff to make for complete privacy of messaging.

## IV-D. Overall Review

Our latency for messaging is, in our belief, sufficiently low as to not inhibit conversation. Our system of acknowledgments and retransmission creates guarantees of reliability within TorText. We believe that ghost messaging does a lot to mask our conversation partners, as well as to some extent when we converse, and is worth the network load tradeoff. We therefore believe that TorText can be evaluated as a viable, private platform for sms.

## References

1. Syverson P, Goldschlag D, Reed M. Anonymous Connections and Onion Routing. Naval Research Labs. 1997
2. https://eprint.iacr.org/2016/1013.pdf
3. http://www.fiercewireless.com/special-report/3g-4g-wireless-network-latency-how-do-verizon-at-t-sprint-and-t-mobile-compare
4. https://pdfs.semanticscholar.org/e5c3/1a99c24ba9ed62e887d2aa4ca4f166987d09.pdf