Some App: Addressing Digital Communication Needs for Vulnerable Communities

Aarushi Dubey
University of Maryland
UID: XXX

December 2023

Abstract

The following paper lays out a design plan and partial theoretical basis for a messaging application mimicked off of Telegram and Signal practices, but with additional qualities to address digital communication needs for vulnerable communities, such as protesters.

1 Introduction

Information security practices used by vulnerable groups for communication are often dependent on what tactics and technologies are available with higher ease-of-use and understanding to a majority. Currently, applications focused on privacy preservation are growing, such as Signal, but are not being as utilized by these marginalized audiences, possibly due to a lack of knowledge, or a fear of who their interactions might be with due to many conservative audiences utilizing these privacy-preserving spaces in the public eye. Groups such as protesters and children exploring gender within conservative households face and may have similar (or unsimilar) security risks and privacy wants, yet many are still unaddressed within digital communication technologies. I propose a messaging service which builds a Telegram-like application with additional capabilities: group/channel-dependent user-choice pseudonymity; big-group random identity swapping; in-message stenography; and geo-fencing turn on/off abilities. A large amount of the theoretical restrictions I detail below are not concrete.

2 Related Work/Current Approaches

Papers such as [1] examine security practices taken on by protesters in the Anti-ELAB Hong Kong protests, finding that protesters utilized messaging platforms such as Telegram or Whatsapp to communicate securely to one another within groups. However, in order to enable and mimic pseudonymity, protesters performed physical acts external to the platform's capabilities, such as utilizing secondary SIM cards or phones. Other papers such as [2] discuss the need to develop technologies that uphold "inclusive security and privacy," a framework that centers marginalized groups and their associated values, wants, and needs during security development. Vulnerable demographics such as homeless people have noted how "untrusting relationships" obstructed users from being "able to trust known people with access to devices & accounts" [3]. [4] focuses on the current issues in making cyber security tools more accessible for marginalized communities. New protocols for pseudonym swapping in the context of vehicle identification for automation have also become recently relevant, and can be useful in analysing how to make pseudonym swapping further private within a communication technology setting [5].

3 Proposed Solution

My proposed solution theorizes a messaging application that builds off of existing Telegram properties, but adds capabilities such as

A user-choice pseudonymity dependent on group/channel expectations;

B big-group random identity swapping;

C in-message stenography;

D and geo-fencing turn on/off abilities.

The proposal focuses on centering (1) user autonomy within group priority, and (2) situational relevance. User autonomy within group priority is defined to be the following: that within a communication technology, users can opt in or out of services offered by a group/channel within the minimum requirements of a group's pre-set security expectations from its members. Situational relevance pertains to the concept that administrators of a group have the abilities to change whether they want to uphold certain group expectations during any given moment. The two concepts work cyclically to allow for changes on both user and administrator sides of communication to ensure continuous group expectation malleability and user autonomy within vulnerable demographic settings. The qualities A through D attempt to adhere to in-group user autonomy and situational relevance at varying strictness levels whilst attempting to maintain privacy and security.

3.1 User-Choice Group Pseudonymity

Users have the ability to choose if they would like a pseudonym generated for their identity per each group or not. So, a user may utilize a chosen name for certain groups, but may opt for pseudonym generation (provided by the messaging platform) for other groups. Administrators may place restrictions on groups to enforce pseudonymity or not upon users' entry or time in group. Users may either choose to adhere to that restriction to participate in the group, or leave. Administrators can change entry settings post group creation, but must uphold new and previous members to new pseudonymity restrictions. In order to build a pseudonym generator for our messaging application, we can utilize Markov chains to develop a pseudonym generator based on either (a) sets of words provided by the group creators, or (b) previously-established sets of words provided by the application itself.

3.2 Random Identity Swapping

A group can enable random pseudonym swapping to occur at some randomized interval, either with only other present pseudonyms in the current group or with a new set of generated pseudonyms. At minimum, the pseudonym swapping will occur on 'display name' level. We can mimic the Signal 'Sealed Sender' protocol, which minimizes the amount of metadata that is accessed during the process in which a message is sent by a given involved server. The 'Sealed Sender' protocol only makes the destination/receiving user and the message timestamp available to the server, which then forgets this information after receiving notice that the destination/receiving user successfully received the message.

3.3 In-Message Stenography

With state actors becoming more and more surveilling of the general population through social media and communication platforms, the need for publicly-accessible hidden or private communication channels is increasing. The goal is to develop a plug-in within a messaging platform that would enable users to hide messages within seemingly normal or relevant conversations within groups. In order to conduct in-message steganography, I utilize the Meteor steganographic algorithm described here [https://meteorfrom.space], which utilizes a sampler for the English language from "generative approximations of natural language" to create its stegotext encodings and decodings based on universal steganography. Specifically, the process works as the following: a context is provided to a generative model, such as GPT-2, which outputs a token dictionary with associated probabilities of which token should follow next. These associated probabilities can be condensed into a cumulative probability distribution, and a mapping of bit strings is done to the distribution, which we will utilize later on. The sender then XOR's a randomly generated cipher with the original plaintext they want to send to produce the bit encoding. Some sequence of x bits from the total bit encoding is then used to index into the mapping over the cumulative probability distribution. The index then produces a stegotext token that is saved, and appended to with the next iteration of sampling. 1

In order for in-message steganography to be curated, rather than providing a random context to begin with into a steganographic technique like Meteor, we can utilize some amount of previous messages sent in a given conversation as our context. The mapped token distribution for the context is then used for just one encoding instance, and for one decoding instance. As the receiver sends their next message, it is added onto the context, and a new token distribution is calculated from the generative model intaking a new context. In order for adversaries or sensors to not be able to recreate the token distributions, the values and number of strings within the next given context length is calculated by the following:

3.3.1 Synthetic Context Requirements

Suppose we have to-be-sender (call them **TBS**), who wants to send a message through in-message steganography, to the to-be-receiver (call them **TBR**). In order for their final stegotext to be as indiscernible as possible from their real messages, the context used to create the stegotext needs to have the following qualities:

A the context should be in the writing style of **TBS** - therefore, in our context, some number of messages sent by **TBS** previously, or some parts of them, should be included.

B **TBS** will be assumedly replying to the decoded content that **TBR** has just sent - therefore, in our context, the content of TBR's last uninterrupted chunk of messages need to be given in the writing style of **TBS**.

C gpt2 can handle at most 1024 tokens - therefore, our context must contain values of (A) and (B) defined above, but be within the token number restriction.

D the context will be created from the stegotext tokens sent previously - not from the raw plaintexts being conveyed across parties.

3.3.2 Synthetic Context Creation Theory

In order to fulfill requirement (A), we can grab a random number of messages that **TBS** has sent that are proximate to the last message **TBS** received. By grabbing a random chunk of messages that **TBS** has sent - rather than grabbing some constant number of recent messages sent by **TBS** - we ensure that an observer cannot learn which messages were used in the context being created in that moment simply by looking at the previous stegotext outputs.

In order to fulfill requirement (B), we can develop (ideally efficient) miniature GPT models that are trained on a person's messages sent. These miniature models can then be used to ask for rewritten versions of **TBR's** previous content but in the style of **TBS**. Such models can be prompted with states such as "what would the voice analysis be for the following passage" and then ask the model to "use the given voice analysis to generate the same content but in the given manner of speaking."

After gathering the outputs from requirements (A) and (B), we can create a proportion of how much information we want to grab from requirements (A) and (B) in order to contain the context being curated within 1024 tokens. Though the exact proportion is not finalized currently, the expected weightage will be given to requirement (B)'s output to maintain the outputted flow of conversation. Finally, the context builds from previous contexts being sent over, not from the underlying plain text communications occurring.

3.3.3 Utilizing Synthesized Context

After **TBS** creates their context, they input their context into the generative sampler to receive & calculate the token cumulative probability distribution, and uses the bit mapping to encode their XOR'ed plaintext (with a pseudo-random mask) into stegotext, **TBS** sends the stegotext through normal messages alongside sending the current context used through key encapsulation (or other methods similar to how symmetric keys are shared). After sending the context, the device of **TBS'** 'forgets' the context and erases it from its

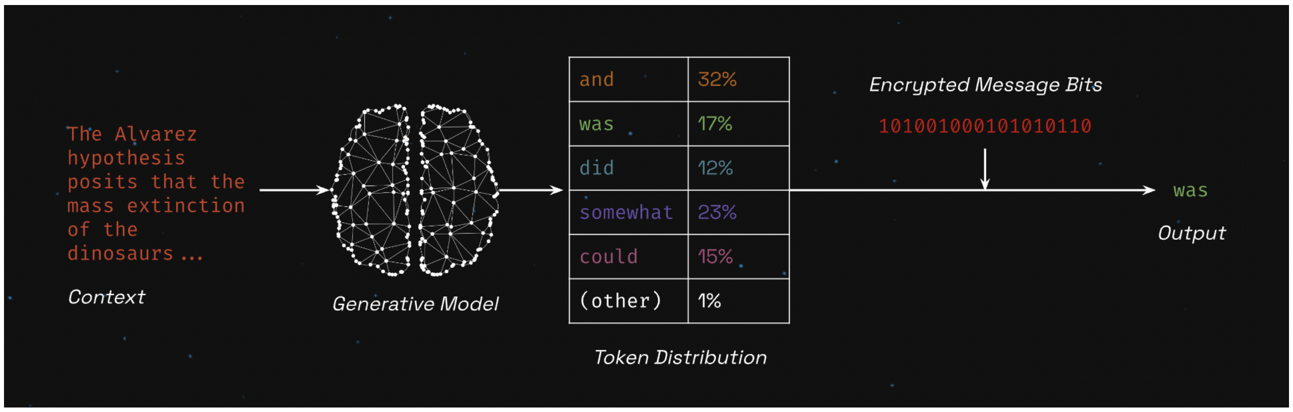

Figure 1: Encoding process of Meteor, as depicted by the creators here https://meteorfrom.space/

'memory.' After **TBR** receives the stegotext and the context, it uses the Meteor decoding scheme to do a probability lookup of a set of bits and a bit-by-bit XOR mask to produce the original plaintext. After doing so, the **TBR** 'forgets' the context it received, and restarts the cycle to send its next stegotext message with roles reversed between **TBR** and **TBS**.

Only the physical devices that the parties are interacting between with have the ability to decode and encode - preventing adversarial listeners from doing retroactive reconstruction due to the pseudo-randomness of the models and contexts involved as well. Though this is a very preliminary stab at in-message steganography, the idea is to expand this communication style to perform efficiently within not just two-people conversations, but between three plus member groups as well.

3.4 GeoFencing Turn On/Off

Groups can have the ability to allow or not allow members to participate in group activities on the platform if they are within a certain geographic radius or not. This ability would be controllable by administrators. If users want to continue participation, they must opt in; otherwise, have their available possible actions be reduced. A use case of this could be if a group of protesters are meeting in person for a protest, and do not want adversarial information being fed into the group from people not physically present at the protest. By turning on geo-fencing, users must verify through the application that they are present in the area of to continue using the group's specified restricted capabilities. Users have the choice to not share their geolocation, and so consequentially their access (either reading, writing, or both) becomes restricted. Administrators may turn off this capability at any time.

4 Conclusion

This paper laid out a possible framework to create a communication application centering marginalized and vulnerable users' agency and preferences. Further development is needed for implementation purposes, specifically regarding how multiple models can be efficiently developed and trained within smaller time-frames. Another application of in-message steganography can be extended to developing device-based text message plugins for privacy-protected messaging, such as through iMessage games/applications development where we can attempt to use in-message steganography to hide our information from our mobile providers.

References

[1] Albrecht, M. R., Blasco, J., Jensen, R. B., & Mareková, L. (2021). Collective Information Security in Large-Scale Urban Protests: the Case of Hong Kong. In 30th USENIX Security Symposium (USENIX Security 21) (pp. 3363-3380).

[2] Wang, Y. (2017, October). The third wave? Inclusive privacy and security. In Proceedings of the 2017 new security paradigms workshop (pp. 122-130).

[3] Sleeper, M., Matthews, T., O'Leary, K., Turner, A., Woelfer, J. P., Shelton, M., ... & Consolvo, S. (2019, May). Tough times at transitional homeless shelters: Considering the impact of financial insecurity on digital security and privacy. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (pp. 1-12).

[4] Renaud, K., Coles-Kemp, L. Accessible and Inclusive Cyber Security: A Nuanced and Complex Challenge. SN COMPUT. SCI. 3, 346 (2022). https://doi.org/10.1007/s42979-022-01239-1

[5] Li, X., Zhang, H., Ren, Y., Ma, S., Luo, B., Weng, J., ... & Huang, X. (2020). PAPU: Pseudonym swap with provable unlinkability based on differential privacy in VANETs. IEEE Internet of Things Journal, 7(12), 11789-11802.

[6] Kaptchuk, G., Jois, T. M., Green, M., & Rubin, A. D. (2021, November). Meteor: Cryptographically secure steganography for realistic distributions. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (pp. 1529-1548).