Veil: The Completely Private Messenger

Research and Initial Plan

December 4, 2021*

Arvid Lunnemark
arvid@mit.edu

ABSTRACT

We outline an initial plan for Veil, and demonstrate its feasibility.

1 INTRODUCTION

Veil will be the world's first completely private messaging platform: if you communicate over Veil, no one will be able to know who you are communicating with. This privacy guarantee will hold even against someone who is able to hack all servers, observe the entire network, and compromise any number of users.

I recommend reading arvid.xyz/complete-privacy [Lun21a] for a motivation of why I think building a completely private messenger is important.

2 RESEARCH

In the research community, completely private messaging often goes under the names *metadata-private communication*, *metadata-hiding communication* or *anonymous communication*. The primary difficulty is guaranteeing privacy while supporting many users with good performance. Dozens of research prototypes have been developed over the past decade; if you're interested in a good overview, I recommend reading this survey by me [Lun21b] (work-in-progress) and this survey by Yossi Gilad [Gil19].

The most promising research prototype is Addra [Ahm+21]. It requires zero trust in servers, and is able to deliver messages with sub-second latency for 32K users. In Addra, senders write their messages to a non-private location on the server, and receivers privately read from that location using private information retrieval (PIR). This happens in synchronized rounds, and to hide patterns, users always send and receive the same amount of data, regardless of whether they actually want to send or receive in a particular round.

There are two major challenges with the PIR approach: (1) keeping network costs down, and (2) keeping server runtime down. Network bandwidth is to some extent fundamental in order to hide patterns, but Addra still has a blowup of around 50x. More recent work, such as OnionPIR [MCR21], which has a network blowup of 4x, might be useful to consider. In terms of server runtime, the PIR approach fundamentally requires $\Omega(n)$ computation time on the server, per use. The user retrievals are completely independent, and can therefore be distributed across many servers, so this is only a problem in terms of server costs. I believe that performance engineering can probably bring down the practical costs of this significantly.

Another challenge is that to receive messages, the messenger client needs to continuously communicate with the server. This is also necessary for privacy: we need data traffic to remain the same regardless of whether the user is actively communicating or not (or else, an adversary can launch timing correlation attacks). This means that it is unlikely to see a perfectly private solution become feasible on platforms with limited power or bandwidth: in particular, **Veil will probably not be feasible on phones**, because phones limit the amount of background computation an app can do.[1]

3 PROPOSED PLAN

For the initial version of Veil, I propose to build a desktop app and a server. The app runs in the background and communicates continuously with the server. Each user will be limited to a small number of friends. Small groupchats may be supported. Because the server costs will be very big, I think we need to charge users a monthly subscription fee to use the app.

3.1 Feasibility

There is a tradeoff between the maximum number of users and how many messages can be processed per time unit. I haven't decided yet which of the following two scenarios to target. Note that each scenario is described in a situation where we assume no performance engineering.

3.2 Scenario 1: 10K users, sub-second latency

In this scenario, we want to create a message service that is competitive with Facebook Messenger, WhatsApp and Signal. We could potentially even support voice calls. For that, we need sub-second latency.

This is very similar to the use case presented in Addra. In their evaluation section, they measured the server CPU time to 22.3 minutes per round for 2^{15} users. The AWS compute rate is around 1 cent per CPU hour, which means that for 32K users, the server cost is roughly $10K per month.

Network-wise, the Addra paper estimates 55 MB network usage per user per five minutes, which adds up to 1 GB per hour. If we were to use OnionPIR instead, network usage would decrease to around 4 MB per hour, which is much more manageable.

3.3 Scenario 2: 1M users, 1-minute latency

With 1 million users, we simply cannot be competitive with other messaging services in terms of usability. Instead, we would have to reimagine how we see messaging: something more akin to email, where a message delay of 1 minute would be okay.

*Last updated: December 4, 2021

[1] One idea is to allow people to create their own servers, to which their phone can connect. This trusted server can constantly communicate with the Veil servers, and can send notifications to the phones. Another idea is to have an app without notifications, and accept that there might be some timing information being leaked.

For 1M users, we would probably want to use OnionPIR or SealPIR. In figure 9 of the SealPIR paper [Ang+18], it is estimated that a single retrieval from a database with 1 million elements takes less than 10 CPU seconds. For one round with 1M retrievals, that becomes around $30 per round. With that, we could do 1 message every minute for a cost of about $1M per month, which would be acceptable with a monthly fee paid by users.

Network costs here would be strictly lower than in the previous scenario, and not something we would need to worry about.

3.4 Risks

(1) The implementation of OnionPIR seems to not exist online. Hopefully we could get it by emailing the authors, but otherwise we would have to implement it based on the paper. Based on this, I think that starting with SealPIR is a good idea, since an implementation of it already exists.
(2) Will people actually pay a monthly fee for a not-so-great messaging service, just because it is completely private?

3.5 Discussion

I personally believe that targeting the 1-minute latency scenario is best. This allows us to start out with relatively low costs, and would enable us to decrease the latency rather than increase it as we grow. I also think that a 1-minute latency is perfectly fine if the app is designed in a way so as to encourage thoughtful, relatively long-form messaging.

Still, we will have a bound of about 1 million users. Given that Tor has around 2 million users per day, this seems fine to a first order. Ideally, of course, the entire world would use this, but so far, that seems to be unfeasible. Further research might be able to find another point in the design space: slightly weaker privacy guarantees, but much better performance.

4 FAQ

Why do Veil as a startup? Tor and Signal, the services closest in spirit to Veil, are both non-profit organizations. Ideally, Veil would also be non-profit, but the server costs are huge. It is simply not possible, sadly, to let people use Veil for free. Therefore, the startup model makes sense.

Why does this not exist yet? Addra was published this year. OnionPIR was published this year. The techniques for doing this simply were not known until very recently.

REFERENCES

[Ahm+21] Ishtiyaque Ahmad et al. "Addra: Metadata-private voice communication over fully untrusted infrastructure". In: *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*. 2021.

[Ang+18] Sebastian Angel et al. "PIR with compressed queries and amortized query processing". In: *2018 IEEE symposium on security and privacy (SP)*. IEEE. 2018, pp. 962–979.

[Gil19] Yossi Gilad. "Metadata-private communication for the 99%". In: *Communications of the ACM* 62.9 (2019), pp. 86–93.

[Lun21a] Arvid Lunnemark. *It Is Time to Move Beyond End-to-End Encryption*. 2021. URL: https://arvid.xyz/posts/it-is-time-to-move-beyond-end-to-end-encryption/ (visited on 11/27/2021).

[Lun21b] Arvid Lunnemark. "Research-in-Progress: Anonymous Communication". In: (2021). URL: https://arvid.xyz/ac-research.

[MCR21] Muhammad Haris Mughees, Hao Chen, and Ling Ren. "OnionPIR: Response Efficient Single-Server PIR". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2021, pp. 2292–2306.