PriFi: Low-Latency Tracking-Resistant Anonymous Communication

Ludovic Barman[*], Mahdi Zamani[†], Italo Dacosta[*], Joan Feigenbaum[†], Bryan Ford[*], Jean-Pierre Hubaux[*], David Wolinsky[‡]

[*]EPFL, [†]Yale University, [‡]Facebook

1 Introduction

Popular anonymity mechanisms such as Tor [?] provide small communication latency, but are vulnerable to *traffic-analysis attacks* that can de-anonymize users. Moreover, known traffic-analysis resistant techniques such as Dissent [?] are impractical for use in latency-sensitive settings such as wireless networks. We propose PriFi, an anonymous communication protocol with provable traffic-analysis resistance and small latency suitable for wireless networks.

PriFi provides a network access mechanism for protecting members of an organization who access the Internet while on-site (via privacy-protected WiFi networking) and while off-site (via privacy-protected virtual private networking or VPN). This provides the members with "local-area" anonymity by preventing the adversary from learning which user within the organization is communicating with the Internet.

PriFi leverages the client-relay-server topology common in WiFi networks to reduce protocol latency significantly. The main novelties in the design of PriFi are **(1)** using a set of servers to pre-compute cryptographic material that cause a major latency bottleneck in the anonymization process when computed on the fly; and **(2)** preventing equivocation attacks by the untrusted relay without adding extra latency.

Network and Threat Model. Consider a set of *clients*, or users who want to access the Internet anonymously. The clients are connected to this network through a *relay*, or router, that can process normal TCP/IP traffic in addition to running our protocol (see Figure **??**). We also introduce a group of servers whose role is to assist the relay in the anonymization process. The servers are off the latency-critical path and may be distributed around the world to maximize trustworthiness.

We consider an adversary who controls a subset of the clients, all but one of the servers, and the relay. The servers are assumed to satisfy the requirements of the *anytrust model* [?]: At least one of the them is honest, and they are all (including the relay) assumed to be available at all times. We assume that all nodes communicate using non-private but authenticated channels; the adversary can observe all such messages when they are sent.

From a user's point of view, PriFi may be viewed as a low-latency VPN service: It receives data from and sends data to the applications running on the user's computer. The relay acts as the other end of the VPN, sending data to the Internet or to the clients. However, unlike traditional VPN services, the relay is not trusted; it may maliciously (possibly by colluding with other untrusted entities) attempt to de-anonymize the clients. The anytrust group of servers will collectively allow PriFi to protect the clients from de-anonymization by the VPN service, without adding latency into the critical communication path.

Background. PriFi builds on earlier work on Dissent [?], which is based on Dining Cryptographer's networks (*DC-nets*) [?] – anonymous communication protocols with provable anonymity against global adversaries. DC-nets anonymizes traffic by asking each participant to transmit equal-length ciphertexts synchronously: One of the ciphertexts contains a message M to be broadcasted and the others simply contain empty/dummy messages. The messages are then combined in such a way that the empty messages cancel out and M is reveled to all participant without disclosing the actual sender.

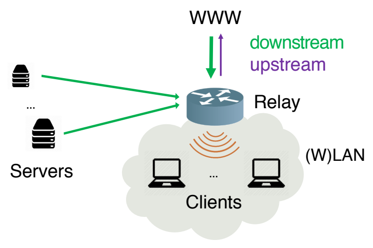

Figure 1: System setup

DC-nets require a reservation mechanism to protect against collisions and a jamming-detection mechanism for preventing denial-of-service attacks. Such mechanisms often reduce the performance of the protocol significantly. Dissent alleviates this by adapting DC-nets to a client-server model in which a small set of anytrust servers assist the clients with the anonymization process. Such a topology, however, has a major impact on latency; for example, it requires several server-to-server rounds of communication to handle client churn, maintain integrity of messages, and ensure accountability of participants. This latency problem associated with the Dissent's topology is what PriFi solves, as it will be described next.

2 Protocol Design

We define *downstream* communication as the data from the Internet to one of the clients, and *upstream* communication as the data from one of the clients towards the Internet. The downstream data is simply broadcasted from the relay to every client by leveraging inexpensive broadcast provided by WiFi networks. The upstream data goes through our anonymization protocol and is sent by the relay towards the Internet. We refer to each time the clients anonymously sending a messages to the relay as a *round*.

We now briefly describe the protocols run jointly by the clients, servers, and the relay in each round for anonymizing an upstream message M sent by one of the clients. These protocols are: *setup*, *scheduling*, and *anonymization*. In the setup protocol, each client agrees with each servers on a shared secret, which is known to both of them but is secret to others. This secret is then used to seed a pseudo-random generator to obtain a stream of pseudo-random bits from which the clients and the servers will compute their ciphertexts.

The scheduling protocol is run in each round to determine which client gets to transmit his message in which round.

The scheduling information needs to remain secret to all entities, as otherwise it can completely break the anonymity of the clients. In PriFi, the servers randomly and verifiably shuffle (using [**?**]) a set of public keys corresponding to the clients. The secret permutation is then sent to all clients each of whom is only able to recognize his own public key in the sequence; other keys look unrelated to anyone without the associated private key.

In the anonymization protocol, every client sends a ciphertext to the relay; one of the ciphertexts contains the message M to be sent to the Internet, and the rest contain empty messages. Every server also sends a ciphertext to the relay. The relay then participates in a distributed protocol jointly with the servers to obtain M from the collected ciphertexts.

The key novelties of PriFi in reducing the latency of Dissent while protecting against an untrusted relay are:

1. Pre-Computing Server Ciphertexts. We observe that the server ciphertexts do not depend on the actual communicated content. Therefore, the servers can compute their ciphertexts ahead of the time before they are needed by the clients and the relay in the actual communication. This eliminates an important latency bottleneck from the critical latency path of the protocol. The servers continuously transmit freshly-produced ciphertexts to the relay throughout a round. Assuming that the servers have a high throughput link to the relay, the arrival of their ciphertext should, in practice, outpace the rate of exchanges being performed by the clients and the relay, reducing protocol latency.
2. Preventing Relay Equivocation. PriFi's client-relay-server topology introduces an important challenge: The untrusted relay can preform equivocation attacks by sending different (inconsistent) downstream messages to the clients to de-anonymize them. For example, in a non-SSL-encrypted communication, the relay can slightly modify the downstream message for each client, and therefore send a unique message to each of them. Then, in the next round, he checks to see what is being requested by one of the clients, and depending on the content requested, he can identify the client. PriFi prevents this without adding extra latency by encrypting clients' upstream messages in such a way that the ciphertext depends on previous downstream message. Therefore, the relay can only decrypt each upstream message if all clients agree on what they received in all past downstream rounds. This allows the clients to ensure they are receiving the same message without imposing the expensive overhead of a consensus protocol.

Other challenges that PriFi will address include:

1. Providing Pseudonymity. Even if an adversary cannot directly link a client to a set of actions, it can run intersection attacks by guessing based upon online clients; especially since members of an organization must identify themselves. PriFi addresses this through pseudonymity provided by anonymous authentication mechanisms such as [**?**], wherein members of an organization prove their membership without divulging their identity.
2. Handling Disruptions. A malicious client can render the channel useless by continuously transmitting in every round causing an untraceable denial-of-service attack. To remove this vulnerability, PriFi relies on a variation of Dissent's accountability mechanism redesigned to be compatible with PriFi's low-latency protocol.
3. Multiple Traffic Groups. Rather than incorporating all clients into the same PriFi communication channel, clients can subscribe to different quality of service channels. Examples of potential categories include web browsing, voice-over-IP, video conferencing, and web streaming. Furthermore, clients that have sufficient bandwidth and low enough latency can participate in as many communication channels as possible but only transmit their cleartext in the appropriate channels. This allows PriFi to provide a strong level of anonymity while supporting clients with different constraints (e.g., battery-powered devices).

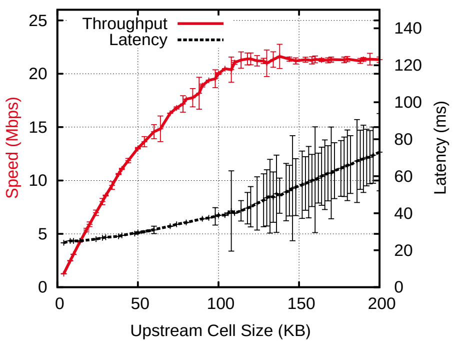

Figure 2: Upstream bandwidth and latency versus cell size

3 Preliminary Results

We implemented a prototype of PriFi in Go. We relied on NIST P-256 elliptic curve for asymmetric cryptographic operations, along with AES 128 bits for symmetric cryptography and SHA-256 as hash function. We deployed our protocol to the Deterlab infrastructure with all nodes running on different machines with the same specifications: 2.4 GHz Intel Xeon X3 processor with 16 GB of RAM. The machines are deployed in two LANs, one for the servers, with a latency of 100 ms, and one for the clients, with a latency of 10 ms. The measured latencies are 106 $\pm$0.5 ms and 16 $\pm$1 ms both over 10 samples, respectively. The relay belongs to both LANs. All network links are 100 Mbps full duplex.

We evaluate the total anonymized upstream throughput when varying the *upstream cell size*, the number of bytes sent to the relay per round and per client. To test the system capacity, we run one client and three servers, and measure the upstream speed this client gets. In addition, we measure the latency as the round-trip time from the client to the relay, and back to the client. As shown in Figure **??**, PriFi is able to anonymize up to about 20 Mbps of traffic with a latency of 40 ms. In another experiment, we used standard network pipelining to reach the maximal throughput of 100 Mbps with almost the same latency.