Katzenpost: Building Privacy Infrastructure in Go

Eva Infeld

BUILDING PRIVACY SOFTWARE IN GO

github.com/katzenpost
evainfeld@riseup.net

Things I will talk about:

- Why? Because surveillance is bad and privacy is essential, and the world is on fire.
- A quick look at our design.
- Tasks we have funding for (that you can help us with)
- Usable code! Example: hybrid post-quantum cryptography

The slides will be dense, for the video's sake. The most important things will be highlighted.

WE NEED TO TALK ABOUT SURVEILLANCE

Realistic modern adversaries are GLOBAL, ACTIVE, SOPHISTICATED and HAVE CONTEXT (+ may have quantum computers)

- Global (and a lot less is enough for statistical analysis)
- Can compromise parts of the network
- Willing to make decisions based on partial evidence and statistical analysis
- Have large computational resources and can do advanced cryptanalysis
- May have quantum computers soon
- Can supplement collected data with rich context of already gathered data on all users from other sources

This is beyond the threat model of leading anonymity systems.

EXAMPLES

Tor doesn't protect against attackers that see both ends of the connection. (But is very good at doing some other things, and the best in its class.)

Neither do VPNs, and they are even vulnerable to someone just watching the VPN, plus they don't have Tor's suite of additional protections and good software. Cover traffic doesn't really change that.

RESISTING SURVEILLANCE

> The only way to protect metadata is to give up **nothing**, because even the smallest correlations become deadly over time.

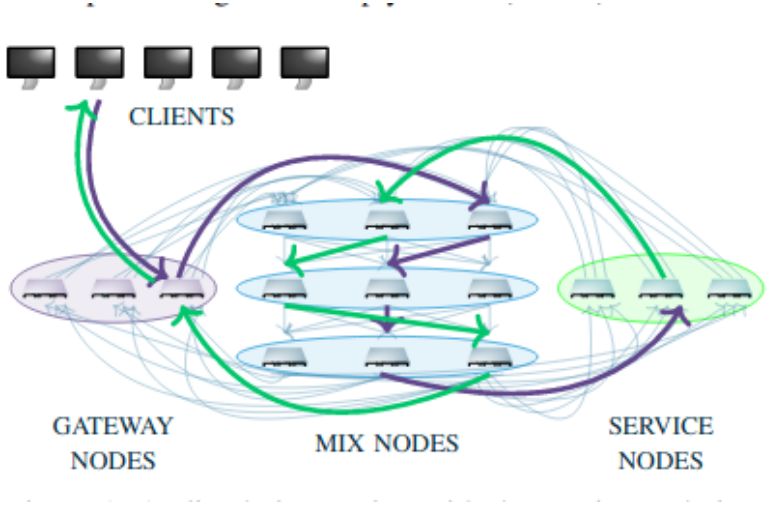
- No message content (we do this with encryption.)
- No social graph data (i.e. who is talking to whom.)
- Hide user location from as many parties as possible, including their contacts.
- No observable traffic patterns

Not an exhaustive list.

This is extremely difficult and every design decision has tade-offs.

OUR DESIGN PRINCIPLES

- Your connection always looks the same.
- All interactions with servers, positioned behind the mix network, are symmetric round-trips whether you're sending or retrieving. (Echos)



- All server interactions are indistinguishable from picking a server at random.
- Persistent relationships between users leak statistical information, minimize it.
- Second party anonymity - your contacts are also seen as a point of failure.
- No forced interactivity. So can't have conventional TCP applications.

WHERE TO FIND OUT MORE

https://arxiv.org/abs/2501.02933

Echomix: a Strong Anonymity System with Messaging

Ewa J. Infeld David Stainton Leif Ryge Threebit Hacker

***Abstract*—Echomix is a practical mix network framework and a suite of associated protocols providing strong metadata privacy against realistic modern adversaries. It is distinguished from other anonymity systems by a resistance to traffic analysis by global adversaries, compromised con-**

We additionally introduce a messaging protocol which is suitable for anonymous group messaging with a realistic threat model, and provides reliability without forcing interactivity. We then present a quantum resistant packet format appropriate for mix networks. Finally, we provide

We will likely rename the whole project to EchoMix soon.

DEVELOPMENT (Go, AGPLv3):
HERE ARE SOME FUNDED NEAR FUTURE TASKS

A mix network, Katzenpost/EchoMix
(https://github.com/katzenpost/katzenpost

1. Implementing new cryptographic protocols for group chat — in progress
2. Implementing new cryptographic protocols for exchanging credentials — in progress
3. Making libraries more modular and accessible — in progress/to do
4. Some design tasks — in progress

A chat client that uses the network, Katzen/Echo
https://github.com/katzenpost/katzen

1. Restructuring the database — almost done
2. Adding attachment functionality — to do
3. Implementing cutting edge audio processing — in progress

nlnet FOUNDATION

W HOLLAND STIFTUNG

SOME BIG PICTURE GOALS

In the longer term, we are still shaping the workflow and you can shape it with us. We have avenues to secure funding for well planned goals.
In the long run we want:

- More audits of the code.
- Mix network software that can be easily instantiated if you want your own network.
- Mix network instance (we have one, Namenlos) that you can easily connect your services to.
- A messenger application that can actually resist realistic adversaries, and has all the functionality and ease of use that today's users expect.
- Easy-to-use-in-your-project code for all the elements we bring to the table: crypto, messaging protocols, media processing tools, network tools, etc.
- Formalizing our designs through a research program, including PIR, evolving communication protocols, improving theoretical foundations.

USE OUR CRYPTO

Hybrid post quantum cryptography combines the security of classical cryptography (usually elliptic curves), and cryptography resistant to known quantum algorithms.

HPQC
hybrid post-quantum cryptography library

signatures
github.com/katzenpost/hpqc/sign

KEM
github.com/katzenpost/hpqc/kem
key encapsulation mechanism

NIKE
github.com/katzenpost/hpqc/nike
non-interactive key exchange
(Diffie-Hellman-style, with public keys)

USE OUR CRYPTO

Some of the cryptographic primitives in HPQC (+ ways to combine them):

NIKE:

- X25519
- X448
- Diffie-Hellman
- CSIDH
- CTIDH
- . . .

KEM:

- All of the above through a NIKE to KEM adapter
- M-KEM
- ML-KEM
- Kyber768
- X-Wing (X25519 + ML-KEM-768)
- . . .

Signatures:

- Ed25519
- SPHINCS+
- . . .

USE OUR CRYPTO

```
import (
        "github.com/katzenpost/hpqc/nike/schemes"
        "github.com/katzenpost/hpqc/nike"
)


func doCryptoStuff() {
        scheme := schemes.ByName("X25519")
        if scheme == nil {
                panic("NIKE scheme not found")
        }

        alicePubKey, alicePrivKey, err := scheme.GenerateKeyPair()
        if err != nil {
                panic(err)
        }

        bobPubKey, bobPrivKey, err := scheme.GenerateKeyPair()
        if err != nil {
                panic(err)
        }

        aliceSharedSecret := scheme.DeriveSecret(alicePrivKey, bobPubKey)
        bobSharedSecret := scheme.DeriveSecret(bobPrivKey, alicePubKey)

        // do stuff with shared secrets.
        // aliceSharedSecret is equal to bobSharedSecret
}
```

TAKEAWAYS

github.com/katzenpost
katzenpost.network (or soon echomix.network)
evainfeld@riseup.net

Takeways:

- Surveillance is bad and privacy is good, and we need stronger anonymity software than we already have.
- Our cool design does things other designs don't.
- We have some funding and welcome your help (but talk to me first!)
- Use our code! It's AGPLv3

Katzenpost/Echomix development has been funded by EU Horizon2020 grant ID: 653497, NLnet, Wau Holland Stiftung, Protocol Labs, Zero Knowledge Network.