PrivateStats: De-identified authenticated logging at scale

Subodh Iyengar
Software Engineer

FACEBOOK

Agenda

01 Motivation

02 Architecture

03 Practical Challenges

04 AB-VOPRF

Client logging is important

Performance

Debugging and monitoring performance of app features, for example, latency of sending a message

Reliability

Debugging and monitoring crashes and errors

Private Logging

Could we log data from the clients without associating it with the user?

Can we trust it?

Are de-identified logs still useful to debug problems?

PrivateStats: Architecture

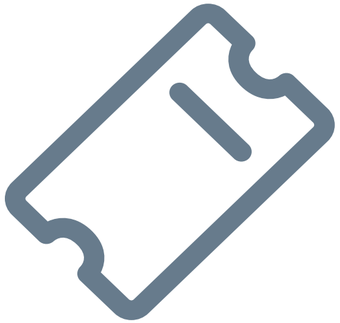
Mobile App

Initiates logging

Application Server

Receives and processes log data

Credential Service



Multi-tenant VOPRF execution service. Manages private keys as well as protects against multiple use of tokens

PrivateStats: Issuing tokens

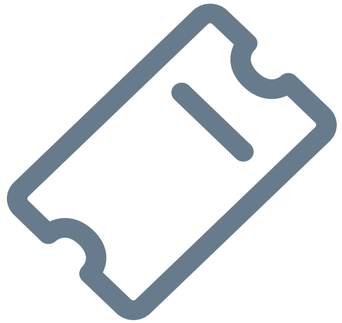
Mobile App

1. Generate a nonce: **X = H(x)**
2. Blind nonce : **M = rG + X**
3. Send the blinded nonce to the application server via an authenticated network call
4. Token = **Z - r * PubKey**
   = **k * X**
5. Verifies the Discrete log proof is correct

Application Server

Authenticates the user

Credential Service



1. Evaluates a VOPRF: **Z = k * M**
2. Returns a zero-knowledge discrete log proof that it evaluated the PRF correctly:
   $\log_G(k * G) == \log_M(k * M)$

PrivateStats: Redeeming tokens

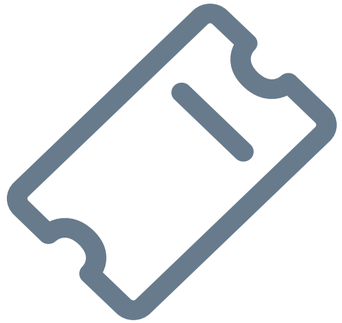
Mobile App

1. App is ready to log a message: **Log**
2. Calculates a **sk = H(X, Token)** from the nonce **X**
3. Calculates a mac on the message: **Mac = HMAC(sk, Log)**
4. Sends **(Log, X, Mac)** to the server via an unauthenticated network call on a separate connection

Application Server

1. Forwards **X** to the credential service
2. Checks whether **HMAC(sk, Log) == Mac** and log data

Credential Service



1. Checks whether **x** has reached usage limit
2. Computes **sk = H(X, kX)** and return this to the application server

Practical Challenges

Client app size

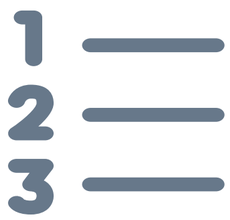
Token reuse

Re-identification resistance

Rate limiting de-identified requests

Practical Challenges

Rate limiting de-identified requests

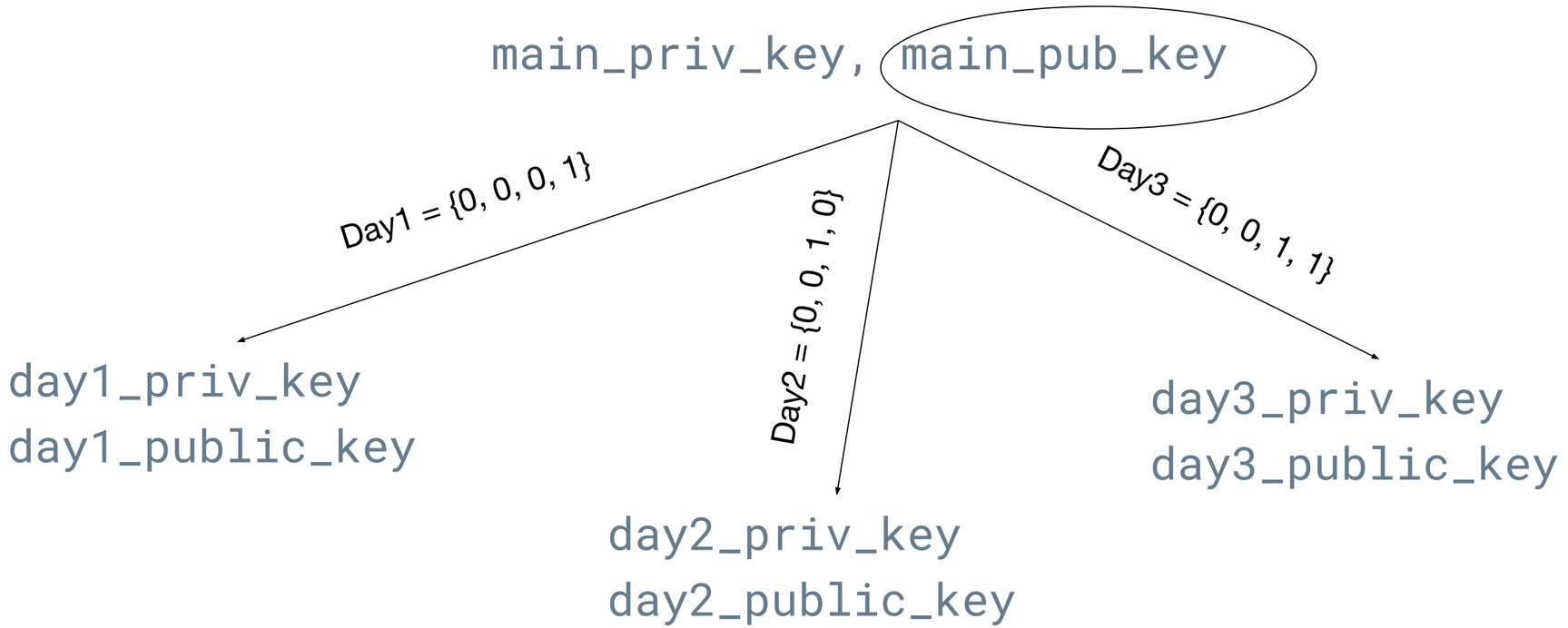


Limit number of tokens per public key per user

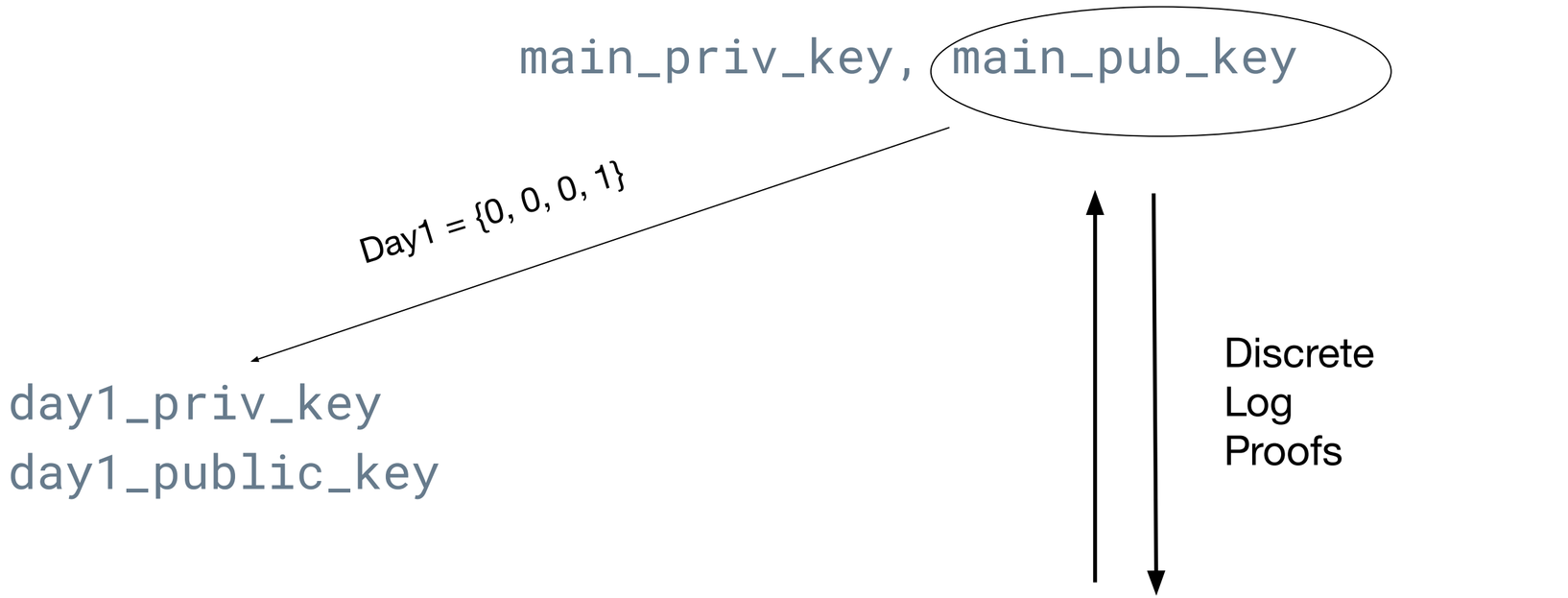
Rotate key.
Rotation makes previous tokens invalid

Key Rotation with Attribute based VOPRFs (AB-VOPRF)

Key Rotation with Attribute based VOPRFs (AB-VOPRF)



Key Rotation with Attribute based VOPRFs (AB-VOPRF)



is day1_public_key correctly derived from main_pub_key?

AB-VOPRF

Pairing-free

Naor-Reingold inspired VOPRF with sequential DLEQ proofs

16 bits: 1.6 ms with proof 0.2ms without proof

We are hoping to bring this to privacy pass with security proofs

Ongoing work

- De-identification can be composed with other methods
- Exploring Querying stored de-identified data via differential privacy
- Using PrivateStats architecture for other applications

https://research.fb.com/privatestats

Summary

A new VOPRF-based system for collecting de-identified user logs tested at scale

Identify and solve practical challenges around token reuse, re-identification resistance, and rate-limiting

Motivate and construct new attribute-based VOPRFs without pairings

Exploring using PrivateStats architecture for non-logging applications

https://research.fb.com/privatestats

Sharon Huang, Subodh Iyengar, Sundar Jeyaraman, Chen-Kuei Lee,
Zutian Luo, Payman Mohassel, Ananth Raghunathan, Shaahid Shaikh,
Yen-Chieh Sung, Albert Zhang

Thank You!