# Decentralized Backup and Recovery of TOTP Secrets

Conor Gilsenan
conorgilsenan@berkeley.edu
UC Berkeley

Noura Alomar
nnalomar@berkeley.edu
UC Berkeley

Andrew Huang
87andrewh@berkeley.edu
UC Berkeley

## ABSTRACT

This work focuses on improving the security, privacy, and usability of backup and recovery methods for apps implementing Time-based One-Time Passwords (TOTP), a widely deployed method of two-factor authentication (2FA). We propose a set of design requirements, explain how several prevalent apps fail to satisfy these requirements, and outline how our scheme leverages decentralized security techniques to satisfy the majority of these requirements.

## CCS CONCEPTS

• **Security and privacy → Multi-factor authentication**; *Usability in security and privacy*.

## KEYWORDS

two factor authentication, 2FA, time-based one-time passwords, TOTP, backup, recovery, usability, security, privacy

**ACM Reference Format:**
Conor Gilsenan, Noura Alomar, and Andrew Huang. 2020. Decentralized Backup and Recovery of TOTP Secrets. In *Proceedings of HoTSoS 2020: Hot Topics in the Science of Security (HoTSoS).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION AND DESIGN GOALS

To authenticate using TOTP [2], client devices use a shared secret to generate a unique one-time password which can be verified by the server. The scheme relies entirely on the shared secret remaining safe and accessible to the client, which often fails in practice. Losing a device, buying a new device, or uninstalling the app are a few examples of how people regularly risk locking themselves out of their accounts protected by TOTP 2FA.

There are dozens of consumer authenticator apps that support the TOTP protocol [6, 8], but they provide mixed support for backing up the TOTP secrets. We believe the following design requirements for a backup and recovery system of TOTP secrets prioritize security, privacy, and usability. Several backup architectures for existing applications rely on a centralized server to facilitate communication and provide short-term or long-term storage; we refer to these service(s) as the *central server*.

**R1: Backup & Recovery.** Users must be able to recover TOTP secrets after losing all personal devices at once.


*HoTSoS, April 07–08, 2020, Lawrence, KS*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

**Table 1: 2FA authenticator apps & design requirements**

| | R1 | R2 | R3 | R4 | R5 |
|---|---|---|---|---|---|
| **Google Authenticator** | – | – | – | – | – |
| **Authy 2FA Authenticator** | ● | – | – | – | – |
| **LastPass Authenticator** | ● | – | – | – | – |
| **Duo Mobile** | ● | ◐ | – | – | – |
| **Microsoft Authenticator** | ● | – | ● | ◐ | – |
| **BLUES 2FA Authenticator** | ● | ● | ● | ● | ◐ |

● = satisfies requirement; ◐ = partially satisfies requirement;
– = does not satisfy requirement;

**R2: No accounts on central server.** Many backup solutions require the user to register an account on the central server. This adds unnecessary friction and often reveals personally identifiable information (PII), such as email addresses and phone numbers. This new account must also be protected by its own password and is typically required to have 2FA enabled. Reintroducing the knowledge factor to protect the backups of the possession factor is somewhat circular logic and presents real usability challenges. A device should be able to start using the system without creating an account on the central server.

**R3: High-entropy keys.** One popular backup architecture is to encrypt TOTP secrets with keys derived from human generated passwords and store the ciphertext on the central server. It is likely that these derived keys will be low entropy because many people choose weak passwords and reuse those passwords, which means the ciphertext would be vulnerable to offline attacks. TOTP secrets and associated metadata should be encrypted using randomly generated high entropy keys.

**R4: Central server cannot access plaintext.** Storing the encryption key and corresponding ciphertext on the same central server greatly diminishes, if not totally negates, the benefit of using high entropy keys. It should be computationally infeasible for the central server to access plaintext secrets.

**R5: No centralized long-term storage.** Using the central server for long-term storage creates a tempting target for attackers hoping to exploit weaknesses and design flaws in the security architecture and represents a single point of failure. The central server cannot lose data that it does not have, so it should not be relied upon for long-term storage.

## 2 ASSESSMENT OF PREVALENT 2FA APPS

This section details the assessment of several prevalent TOTP 2FA authenticator apps against the proposed requirements; Table 1 presents a summary.

**Google Authenticator** [7] does not offer any backup capability by design, so fails all requirements.

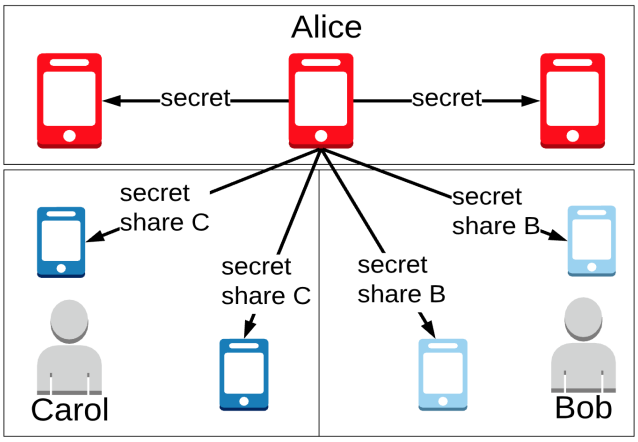



**Figure 1: Adding a new TOTP secret; Alice has three paired personal devices (red) and her Trusted Friends, Carol and Bob, each have two paired personal devices (blues).**

**Authy 2FA Authenticator** [4] uses AES-256 to locally encrypt TOTP secrets using an encryption key derived from a password (violates **R3** and **R4**) and stores the resulting ciphertext in the Authy cloud service (violates **R5**). A valid phone number is required to use the Authy app, which in essence is a user account (violates **R2**).

**LastPass Authenticator** [3] stores TOTP secrets in the user's existing LastPass vault (violates **R2**). The vault itself is encrypted locally with a key derived from a password (violates **R3** and **R4**) and the ciphertext is stored on LastPass servers (violates **R5**).

**Duo Mobile** [1] locally encrypts TOTP secrets using an encryption key derived from a password (violates **R3** and **R4**) and stores the resulting ciphertext in Google Drive or iCloud (violates **R5**). We say that R2 is partially satisfied because, even though a valid Google or iCloud account is required, we concede that Duo leverages the user's existing accounts and does not require a new account on a Duo cloud service.

**Microsoft Authenticator** [9] satisfies **R3** by encrypting TOTP secrets using a randomly generated AES 256-bit keys obtained from a Microsoft key service. Ciphertext is stored in one of two central places (violates **R5**): iCloud for iOS devices and a Microsoft storage service for Android devices. Since Microsoft can access the plaintext of Android backups[1] (they have both the key and ciphertext), but cannot access the plaintext of iOS backups (they cannot access the ciphertext), we say that **R4** is partially satisfied. An account is required to access the key- and storage- services[2] (violates **R2**).

## 3 BLUES 2FA AUTHENTICATOR APP

In this work, we present Blues 2FA, a protocol for the decentralized backup and recovery of TOTP secrets that provides stronger security and privacy guarantees and, we argue, a more compelling user experience than existing alternatives.

Users securely pair personal devices by scanning a QR code, as outlined in the Krypton secure pairing protocol [5]. Blues 2FA syncs full TOTP secrets across all personal devices in real-time, which allows users with multiple devices to *self-recover* if only a single device is lost.

Users can also securely add Trusted Friends[3] by scanning a QR code on the friend's device. Each full TOTP secret is encrypted with a random key and shares of that key are created using Shamir's secret sharing [10]. All devices of each Trusted Friend is sent the ciphertext and a share of the corresponding key. This prevents an individual Trusted Friend from generating valid OTPs, but allows the user to contact a configured subset of Trusted Friends to *socially recover* if they lose all personal devices at once. Figure 1 details the process of adding a new TOTP secret.

It is possible to manually sync devices via QR code, but we leverage a central server to enable real-time syncing between devices. This protocol satisfies **R3** by encrypting TOTP secrets and associated metadata with high-entropy keys, which are randomly generated on client devices. All messages sent through the central server are end-to-end encrypted and authenticated, and keys never leave client devices (satisfies **R4**). An account is not required to begin communicating via the central server (satisfies **R2**). In the majority of cases, the central server only holds messages for a brief time until they are retrieved by the intended recipient. However, messages queued for devices that are offline for an extended period could be lost before they are successfully retrieved, so we mark **R5** as only partially satisfied.

## 4 PILOT STUDY DESIGN AND FUTURE WORK

We implemented the user experience dictated by the Blues 2FA protocol and conducted a pilot of an in-lab survey driven study. Users enable TOTP 2FA on a Dropbox account and log in/out several times while completing a series of tasks which exercise *self-recovery* and *social recovery* functionality. We formed a concrete list of necessary improvements to the prototype before replicating this in-lan study at full scale. We also want to design a longitudinal study in a more realistic environment.

## REFERENCES

[1] [n.d.]. *Duo Restore - Guide to Two-Factor Authentication.* Retrieved December 13, 2019 from https://guide.duo.com/duo-restore
[2] 2011. *TOTP: Time-Based One-Time Password Algorithm.* Retrieved October 02, 2019 from https://tools.ietf.org/html/rfc6238
[3] 2017. *Announcing Cloud Backup for LastPass Authenticator: Easier multifactor security for everyone.* Retrieved December 13, 2019 from https://blog.lastpass.com/2017/05/announcing-cloud-backup-for-lastpass-authenticator-easier-multifactor-security-for-everyone.html
[4] 2018. *How Authy 2FA Backups Work.* Retrieved December 13, 2019 from https://authy.com/blog/how-the-authy-two-factor-backups-work/
[5] 2018. Our Zero-Trust Infrastructure. Retrieved Fenruary 28, 2020 from https://krypt.co/blog/posts/krypton-our-zero-trust-infrastructure.html
[6] 2019. *Apple App Store Search Results - "authenticator".* Retrieved October 02, 2019 from https://www.apple.com/us/search/authenticator?src=serp
[7] 2019. *Google Authenticator for Android (Open Source Version).* Retrieved December 13, 2019 from https://github.com/google/google-authenticator-android/blob/efac95c88ef8d9f8be3c887fbcd2c2fdf4f45dbe/README.md
[8] 2019. *Google Play Store Search Results - "2FA authenticator".* Retrieved October 02, 2019 from https://play.google.com/store/search?q=2fa%20authenticator&c=apps&hl=en
[9] 2019. *How it works: Backup and restore for Microsoft Authenticator.* Retrieved February 28, 2020 from https://techcommunity.microsoft.com/t5/Azure-Active-Directory-Identity/How-it-works-Backup-and-restore-for-Microsoft-Authenticator/ba-p/1006678
[10] Adi Shamir. 1979. How to share a secret. *Commun. ACM* 22, 11 (1979), 612–613.

[1] https://twitter.com/Alex_T_Weinert/status/1195822117922562054
[2] https://twitter.com/Alex_T_Weinert/status/1195856795773751296

[3] Trusted Friends are people known to the user in real life, such as a family member or friend.

# *Decentralized Backup & Recovery of TOTP Secrets*

**Conor** Gilsenan
conorgilsenan@berkeley.edu
@conorgil

## Background

- Time-based One-Time Passwords (TOTP) is a widely deployed 2FA method
  - Client uses a shared secret to generate a unique OTP, which the server validates
- People lose access to the secret and risk locking themselves out of their accounts:
  - lost/new phones
  - uninstall the app
- Backup/recovery in prevalent 2FA apps have security, privacy, and usability issues

## Our solution

- User can s*elf-recover* if one of many personal devices is lost
  - Full TOTP secrets are synced across paired personal devices
- User can *socially recover* from Trusted Friends when all personal devices are lost
  - Devices of Trusted Friends receive encrypted TOTP secrets and a share of the corresponding keys (i.e. Shamir's)
- Messages between devices are end-to-end encrypted and authenticated

## Pilot study design

- We designed an in-lab, survey driven study
- Users are asked to enable TOTP 2FA on a Dropbox account and log in/out during several controlled scenarios

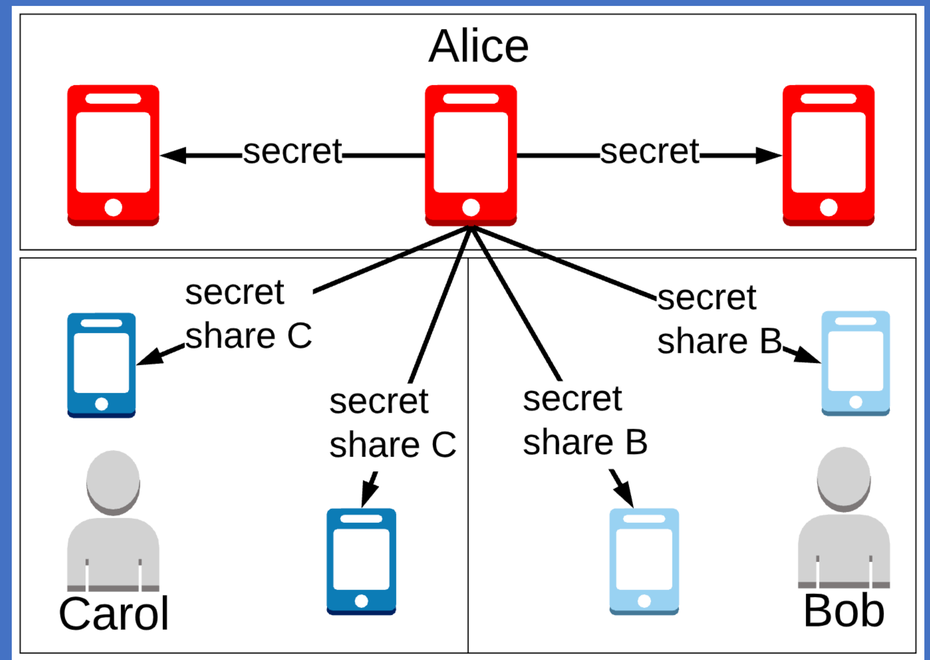
## Future work

- Security analysis of prevalent 2FA TOTP apps to detail risks of backup methods
- Update prototype based on learnings from pilot study and replicate study at full scale
- Design and execute longitudinal study focused on *social recovery*

We designed a decentralized backup and recovery scheme for TOTP secrets that provides stronger security and privacy guarantees than existing 2FA apps.

1) self recovery

2) social recovery




Adding a new TOTP secret; Alice has three paired personal devices (red) and her Trusted Friends, Carol and Bob, each have two paired personal devices (blues).

Take a picture to download the poster abstract

## Presenter Notes

### Design requirements

- R1: Backup & Recovery
- R2: No accounts on central server
- R3: High-entropy keys
- R4: Server cannot access plaintext
- R5: No centralized long-term storage

TABLE I
DESIGN REQUIREMENTS APPLIED TO 2FA AUTHENTICATOR APPS

| | R1 | R2 | R3 | R4 | R5 |
|---|---|---|---|---|---|
| **Google Authenticator** | – | – | – | – | – |
| **Authy 2FA Authenticator** | ● | – | – | – | – |
| **LastPass Authenticator** | ● | – | – | – | – |
| **Duo Mobile** | ● | ◐ | – | – | – |
| **Microsoft Authenticator** | ● | – | ● | ◐ | – |
| **BLUES 2FA Authenticator** | ● | ● | ● | ● | ◐ |

● = satisfies requirement; ◐ = partially satisfies requirement; – = does not satisfy requirement;

### Krypton secure pairing protocol

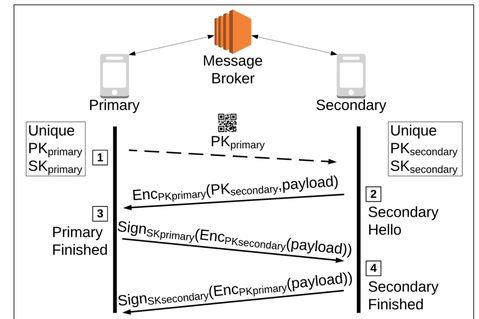



Fig. 1. The Krypton secure pairing protocol for exchanging public keys between two devices.

### Other

- Generating TOTP secret shares

```
let kt = random_256_bit_key()
let cipher_text = encrypt(kt, s, metadata)
let shares = shamirs_secret_sharing(kt)
```

  - Each device of TF[i] receives cipher_text and shares[i]
- Server can verify message deletion via a simple commitment scheme
- Heartbeats can provide confidence in availability of all secret shares
- Even a user with a single personal device can use *social-recovery*
- Device revocation only protects future TOTP secrets; rotate existing secrets

Conor Gilsenan
Noura Alomar
Andrew Huang

BLUES BERKELEY LABORATORY FOR USABLE AND EXPERIMENTAL SECURITY