Security and Privacy Failures in Popular 2FA Apps

Conor Gilsenan  
UC Berkeley / ICSI

Fuzail Shakir  
UC Berkeley

Noura Alomar  
UC Berkeley

Serge Egelman  
UC Berkeley / ICSI

ConorGilsenan@berkeley.edu

AllThingsAuth.com/totp-apps
TOTP 2FA

time-based one-time passwords
Google Authenticator

Google (surfingfan@gmail.com)
464 614

Google (hikingfan@gmail.com)
436 232
otpauth://totp/alice@example.com?secret=SomeSecret&issuer=SomeCompany

Alice's email address or username

The shared secret

The service provider
otpauth://totp/alice@example.com?secret=SomeSecret&issuer=SomeCompany

Alice's email address or username

The shared secret

The service provider
No TOTP secret? No OTPs to log in! 😱
No TOTP secret? No OTPs to log in!

TOTP apps have backup mechanisms! 😎
No TOTP secret? No OTPs to log in!

TOTP apps have backup mechanisms!

Impacts to security & privacy? 😐
No TOTP secret? No OTPs to log in!

TOTP apps have backup mechanisms!

Impacts to security & privacy?

Understudied, so we found out!
Research Questions
1) What **personal info**, if any, is **leaked** when using TOTP backups?
1) What personal info, if any, is leaked when using TOTP backups?

2) What is the risk of an attacker obtaining a TOTP backup?
1) What **personal info**, if any, is **leaked** when using TOTP backups?

2) What is the **risk of an attacker obtaining** a TOTP backup?

3) What is the **risk of an attacker compromising** the TOTP secret(s) stored within an obtained TOTP backup?
Methods
22 TOTP apps

- 100k+ installs
- backup mechanism
1) **Record traffic (after decrypting TLS)**
1) Record traffic (after decrypting TLS)

2) Cryptanalysis (reverse-engineer)
1) **Record traffic** (after decrypting TLS)
2) **Cryptanalysis** (reverse-engineer)
3) **Verify** (prove it)
Key Findings
Backup Mechanisms

# of apps

See paper Table 1
Backup Mechanisms

# of apps

QR Code

See paper Table 1
Backup Mechanisms

See paper Table 1
Backup Mechanisms

- QR Code
- Cloud Sync
- File Export

See paper Table 1
Backup Mechanisms

<table>
<thead>
<tr>
<th># of apps</th>
<th>QR Code</th>
<th>Cloud Sync</th>
<th>File Export</th>
<th>Sharing</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>11</td>
<td>7</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

See paper Table 1
Backup Mechanisms

<table>
<thead>
<tr>
<th># of apps</th>
<th>QR Code</th>
<th>Cloud Sync</th>
<th>File Export</th>
<th>Sharing</th>
<th>Android Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td></td>
<td>11</td>
<td>7</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

See paper Table 1
Backup Mechanisms

Minimum Install Count

- QR Code: 101 (78.8 million)
- Cloud Sync: 2 (2 million)
- File Export: 2 (2 million)
- Sharing: 2.2
- Android Backup: 2.2

See paper Table 1
Backup Mechanisms

Minimum Install Count

Installs (millions)

- QR Code
- Cloud Sync
- File Export
- Sharing
- Android Backup

See paper Table 1
Account Recovery Conundrum

- passwords
- SMS
- email

Photo by Benoit Beaumatin on Unsplash
SMS is dead! Long live SMS!

4 apps relied *only* on SMS to authenticate the user during recovery
No Encryption

2 apps sent plaintext TOTP secrets to the app developers
Encrypted Backups

- 15 apps supported encryption
- Most had serious crypto flaws
How are keys generated?
Keys Derived From Passwords
Microsoft Authenticator

Microsoft Corporation

50+ million installs
Weak Password = Weak Key
Severely Inadequate Password Policies

![Bar Chart](chart.png)

# of apps

Min Password Length

See paper Table 3
Severely Inadequate Password Policies

![Bar Chart: Minimum Password Length vs Number of Apps]

See paper Table 3
Weak Key Derivation

Algo count (n=14)

See paper Table 3
Weak Key Derivation

Algo count (n=14)

See paper Table 3
Weak Key Derivation

Algo count (n=14)

See paper Table 3
Weak Key Derivation

Weak PBKDF2 configurations
\(\text{min} = 10k, \text{median} = 10k, \text{max} = 160k\)

See paper Table 3
Where do keys go?
Poor Key Management

4 apps sent the ciphertext and key (or password from which it was derived) to the app developers.
Microsoft Authenticator

Microsoft Corporation

50+ million installs
1. Request Key
2. Receive Key
3. Android ENC_{key}(TOTP secret)

Microsoft Key Server

Microsoft Storage

Alice

Trusted Device
How are keys used?
Encryption Algos

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># of apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES-ECB</td>
<td>3</td>
</tr>
<tr>
<td>AES-CBC</td>
<td>5</td>
</tr>
<tr>
<td>AES-GCM</td>
<td>3</td>
</tr>
<tr>
<td>Xsalsa20_Poly1305</td>
<td>2</td>
</tr>
</tbody>
</table>
Privacy Issues
Private Info Disclosed in Backups

Some apps encrypted *only* the TOTP secret. Sent the TOTP issuer and username in plaintext.
Recommendations
Encrypt all TOTP fields
(username, secret, website name)

otpauth://totp/alice@example.com?secret=SomeSecret&issuer=SomeCompany
Deriving Keys From Passwords

1) Encourage strong pwds
Deriving Keys From Passwords

1) Encourage strong pwds

2) *ALWAYS* keep password local to app
Deriving Keys From Passwords

1) Encourage strong pwds
2) *ALWAYS* keep password local to app
3) Store derived key in Android Key Store
Deriving Keys From Passwords

1) Encourage strong pwds
2) *ALWAYS* keep password local to app
3) Store derived key in Android Key Store
4) Use Argon2 as KDF
time(key derivation) >= 30 sec
Responsible Disclosure
Questions, please!

ConorGilsenan@berkeley.edu
AllThingsAuth.com/totp-apps

blues-lab / totp-app-analysis-public Public

Security and Privacy Failures in Popular 2FA Apps

★ 9 stars   ★ 0 forks   📈 Activity
Backup Slides
Follow-on work
1) TOTP backup mechanisms:
   a) Do users actually utilize them?
   b) Do they actually help users avoid account lockout?

2) Personal info leaked via TOTP backup mechanisms:
   a) Are users aware they are sharing this info?
   b) Are users comfortable sharing this info?
Tables
<table>
<thead>
<tr>
<th>Abbreviated Name</th>
<th>APK id@version</th>
<th>Installs</th>
<th>QR Codes</th>
<th>Cloud Sync</th>
<th>File Export</th>
<th>Sharing</th>
<th>Android Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Plaintext</td>
<td>Encrypted</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Encrypted</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google Authenticator</td>
<td>com.google.android.apps.authenticator2@v5.10</td>
<td>100M+</td>
<td>Y</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Authenticator</td>
<td>com.azure.authenticator@v6.2204.2757</td>
<td>50M+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duo Mobile</td>
<td>com.duosecurity.duomobile@v4.15.0</td>
<td>10M+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twilio Authy</td>
<td>com.authy.authy@v24.8.5</td>
<td>10M+</td>
<td>-</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Latch</td>
<td>com.elevenpaths.android.latch@v2.2.4</td>
<td>5M+</td>
<td>-</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LastPass Authenticator</td>
<td>com.lastpass.authenticator@v2.5.0</td>
<td>1M+</td>
<td>-</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2FAS</td>
<td>com.twofasapp@v3.11.0</td>
<td>1M+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yandex.Key</td>
<td>ru.yandex.key@v2.7.0</td>
<td>1M+</td>
<td>-</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FreeOTP Authenticator</td>
<td>org.fedorahosted.freeotp@v1.5</td>
<td>1M+</td>
<td>-</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authenticator</td>
<td>com.pixiplicity.auth@v1.0.6</td>
<td>500k+</td>
<td>Y</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Salesforce Authenticator</td>
<td>com.salesforce.authenticator@v3.8.5</td>
<td>500k+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code Generator</td>
<td>net.codemonkey.otpgeneratorapp@v6.1</td>
<td>500k+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOTP Authenticator</td>
<td>com.authenticator.authservice2@v1.89</td>
<td>100k+</td>
<td>Y</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aegis Authenticator</td>
<td>com.beendevdevelopment.aegis@v2.0.3</td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Auth0 Guardian</td>
<td>com.auth0.guardian@v1.5.3</td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>App Authenticator</td>
<td>authentic.your.app.authenticator@v1.5</td>
<td>100k+</td>
<td>Y</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>andOTP</td>
<td><a href="mailto:org.shadowicer.flocke.andotp@v0.9.0.1-play">org.shadowicer.flocke.andotp@v0.9.0.1-play</a></td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y^</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zoho OneAuth</td>
<td>com.zoho.accounts.oneauth@v2.1.0.5</td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authenticator Pro</td>
<td>me.jmh.authenticator@v1.15.10</td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAAPASS</td>
<td>com.solidpass.saaqpass@v2.2.28</td>
<td>100k+</td>
<td>-</td>
<td></td>
<td>Y</td>
<td>Y*</td>
<td></td>
</tr>
<tr>
<td>Authentic Password</td>
<td>authentic.password.authenticator@v1.3</td>
<td>100k+</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Y*</td>
<td></td>
</tr>
<tr>
<td>Mobile Authenticator</td>
<td>authenticator.mobile.authenticator@v1.7</td>
<td>100k+</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Y*</td>
<td></td>
</tr>
<tr>
<td><strong>TOTAL apps</strong></td>
<td></td>
<td>181.5M+</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td><strong>TOTAL installs</strong></td>
<td></td>
<td>101M+</td>
<td>6.1M+</td>
<td>73.7M+</td>
<td>1.8M+</td>
<td>1.5M+</td>
<td>2M+</td>
</tr>
</tbody>
</table>

Table 1: Overview of the backup mechanisms supported in each app. Y* indicates that there is a serious security flaw in the implementation and/or usage of cryptography (see Section 5.3). Y^ indicates support for multiple types of encrypted file exports (see Section 5.3.4). Values in parentheses were obtained from documentation and observation only (see Section 6.4).
<table>
<thead>
<tr>
<th>Abbreviated Name</th>
<th>Encrypted?</th>
<th>PII to use cloud backups</th>
<th>Backup Location</th>
<th>TOTP Data Leaked</th>
<th>Obtain Backup With...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>phone</td>
<td>email</td>
<td>name</td>
<td>dob</td>
</tr>
<tr>
<td>Microsoft Authenticator</td>
<td>Yes*</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Duo Mobile</td>
<td>Yes</td>
<td>-</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
</tr>
<tr>
<td>Twilio Authy</td>
<td>Yes</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Latch</td>
<td>No</td>
<td>-</td>
<td>Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LastPass Authenticator</td>
<td>(Yes)</td>
<td>-</td>
<td>Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2FAS</td>
<td>No</td>
<td>-</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
</tr>
<tr>
<td>Yandex.Key</td>
<td>Yes*</td>
<td>Y</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Salesforce Authenticator</td>
<td>Yes*</td>
<td>Y</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TOTP Authenticator</td>
<td>Yes*</td>
<td>-</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
</tr>
<tr>
<td>Zoho OneAuth</td>
<td>Yes*</td>
<td>-</td>
<td>Y</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SAASPASS</td>
<td>No</td>
<td>Y</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Overview of the backup mechanisms that automatically sync data to the cloud. Yes* indicates a serious security flaw in the implementation and/or usage of cryptography (see Section 5.3). Y^ indicates the field is conditionally included in the backup as plaintext (see Section 5.5). Values in parentheses were obtained from documentation and observation only (see Section 6.4).
<table>
<thead>
<tr>
<th>Abbreviated Name</th>
<th>Key Source</th>
<th>Password Min Len</th>
<th>KDF and Configuration</th>
<th>KDF Salt</th>
<th>Encryption Algorithm</th>
<th>Cipher Text Integrity</th>
<th>Decryption Heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Authenticator</td>
<td>Random*</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>AES-128-CBC</td>
<td>HMAC-SHA256</td>
<td>n/a</td>
</tr>
<tr>
<td>Zoho OneAuth</td>
<td>Password*</td>
<td>3</td>
<td>SHA-256 i = 1</td>
<td>none</td>
<td>AES-256-ECB</td>
<td>none</td>
<td>Base32</td>
</tr>
<tr>
<td>Salesforce Authenticator</td>
<td>Password*</td>
<td>4</td>
<td>PBKDF2-HMAC-SHA256 i = 10,000</td>
<td>random</td>
<td>AES-256-CBC</td>
<td>none</td>
<td>JSON</td>
</tr>
<tr>
<td>Yandex.Key</td>
<td>Password*</td>
<td>6</td>
<td>scrypt N = 2^15, r = 20, p = 1</td>
<td>random</td>
<td>Xsalsa20_Poly1305</td>
<td>AEAD</td>
<td>n/a</td>
</tr>
<tr>
<td>TOTP Authenticator</td>
<td>Password</td>
<td>8</td>
<td>SHA-256 i = 1</td>
<td>none</td>
<td>AES-256-CBC</td>
<td>none</td>
<td>JSON</td>
</tr>
<tr>
<td>Authenticator</td>
<td>Password</td>
<td>10</td>
<td>PKCS12-SHA256 i = 65,536</td>
<td>hard coded</td>
<td>AES-256-ECB</td>
<td>none</td>
<td>URI</td>
</tr>
<tr>
<td>App Authenticator</td>
<td>Password</td>
<td>10</td>
<td>PKCS12-SHA256 i = 65,536</td>
<td>hard coded</td>
<td>AES-256-ECB</td>
<td>none</td>
<td>URI</td>
</tr>
<tr>
<td>Auth0 Guardian</td>
<td>Password</td>
<td>1</td>
<td>(PBKDF2-HMAC-SHA1 i = 10,000)</td>
<td>(random)</td>
<td>(AES-256) (HMAC)</td>
<td>(n/a)</td>
<td></td>
</tr>
<tr>
<td>Authenticator Pro</td>
<td>Password</td>
<td>1</td>
<td>PBKDF2-HMAC-SHA1 i = 64,000</td>
<td>random</td>
<td>AES-256-CBC</td>
<td>none</td>
<td>JSON</td>
</tr>
<tr>
<td>2FAS</td>
<td>Password</td>
<td>1</td>
<td>PBKDF2-HMAC-SHA256 i = 10,000</td>
<td>random</td>
<td>AES-256-GCM</td>
<td>AEAD</td>
<td>n/a</td>
</tr>
<tr>
<td>Aegis Authenticator</td>
<td>Password</td>
<td>2</td>
<td>scrypt N = 2^15, r = 8, p = 1</td>
<td>random</td>
<td>AES-256-GCM</td>
<td>AEAD</td>
<td>n/a</td>
</tr>
<tr>
<td>andOTP</td>
<td>Password</td>
<td>4</td>
<td>PBKDF2-HMAC-SHA1 i = [140,000 - 160,000]</td>
<td>random</td>
<td>AES-256-GCM</td>
<td>AEAD</td>
<td>n/a</td>
</tr>
<tr>
<td>Twilio Authy</td>
<td>Password</td>
<td>6</td>
<td>PBKDF2-HMAC-SHA1 i = 10,000</td>
<td>random</td>
<td>AES-256-CBC</td>
<td>none</td>
<td>Base32</td>
</tr>
<tr>
<td>Duo Mobile</td>
<td>Password</td>
<td>10</td>
<td>argon2i m = 128 Mb, t = 6, p = 1</td>
<td>random</td>
<td>Xsalsa20_Poly1305</td>
<td>AEAD</td>
<td>n/a</td>
</tr>
<tr>
<td>LastPass Authenticator</td>
<td>Password</td>
<td>12</td>
<td>(PBKDF2-HMAC-SHA256 i = 100,100)</td>
<td>(random)</td>
<td>(AES-256) (HMAC)</td>
<td>(n/a)</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Cryptographic details of app backup mechanisms. The asterisk (*) indicates that the app leaks the encryption key and/or password to the same service which stores the ciphertext, allowing that service to decrypt the TOTP backup (see Section 5.3.3). Square brackets indicate the min and max of a range, inclusive. Values in parentheses were obtained from documentation and observation only (see Section 6.4). The abbreviations for KDF configurations are: SHA/PKCS12/PBKDF2 (i = iterations), scrypt (N= CPU/memory cost, r = block size, p = parallelism), and Argon2 (m = memory, t = time/iterations, p = parallelism).