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

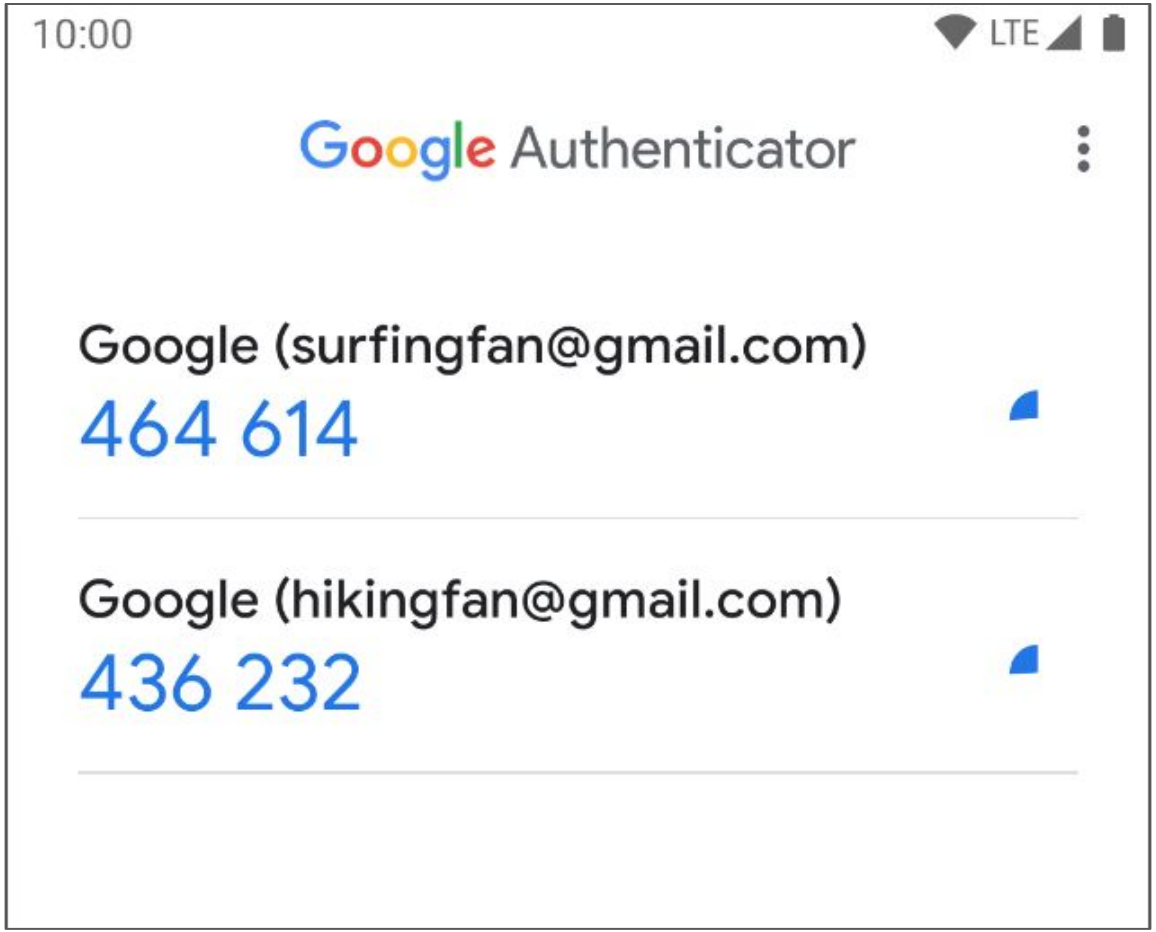
32ND USENIX SECURITY SYMPOSIUM

AUGUST 9–11, 2023
ANAHEIM, CA, USA

BLUES
BERKELEY LABORATORY FOR USABLE AND EXPERIMENTAL SECURITY

TOTP 2FA

time-based one-time passwords



https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2&hl=en_US&gl=US

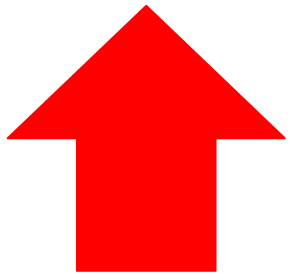

Scan to setup TOTP

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

Alice's email address or username

The **shared secret**

The service provider

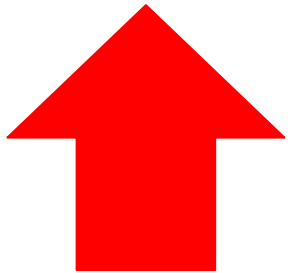


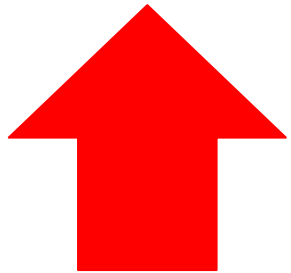
otpauth://totp/**alice@example.com**?secret=**SomeSecret**&issuer=**SomeCompany**

Alice's email address or username

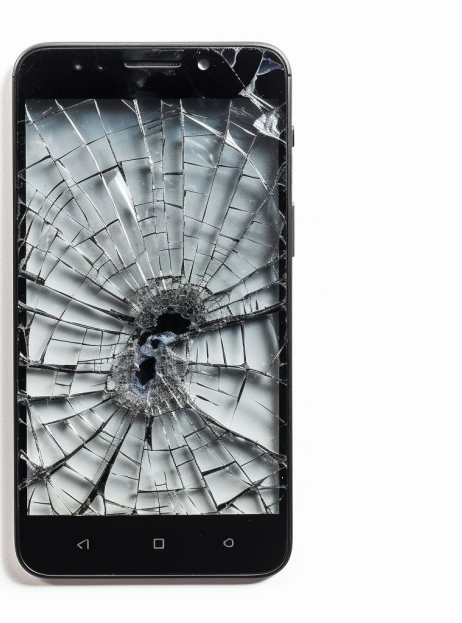
The **shared secret**

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)

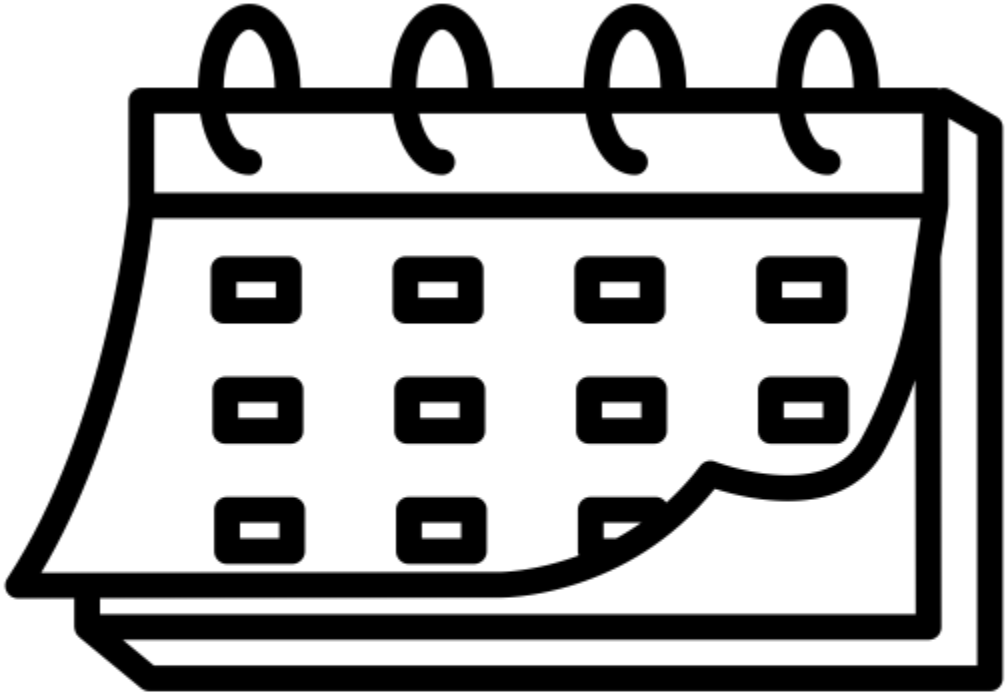
2) **Cryptanalysis** (reverse-engineer)

1) **Record traffic** (after decrypting TLS)

2) **Cryptanalysis** (reverse-engineer)

3) **Verify** (prove it)

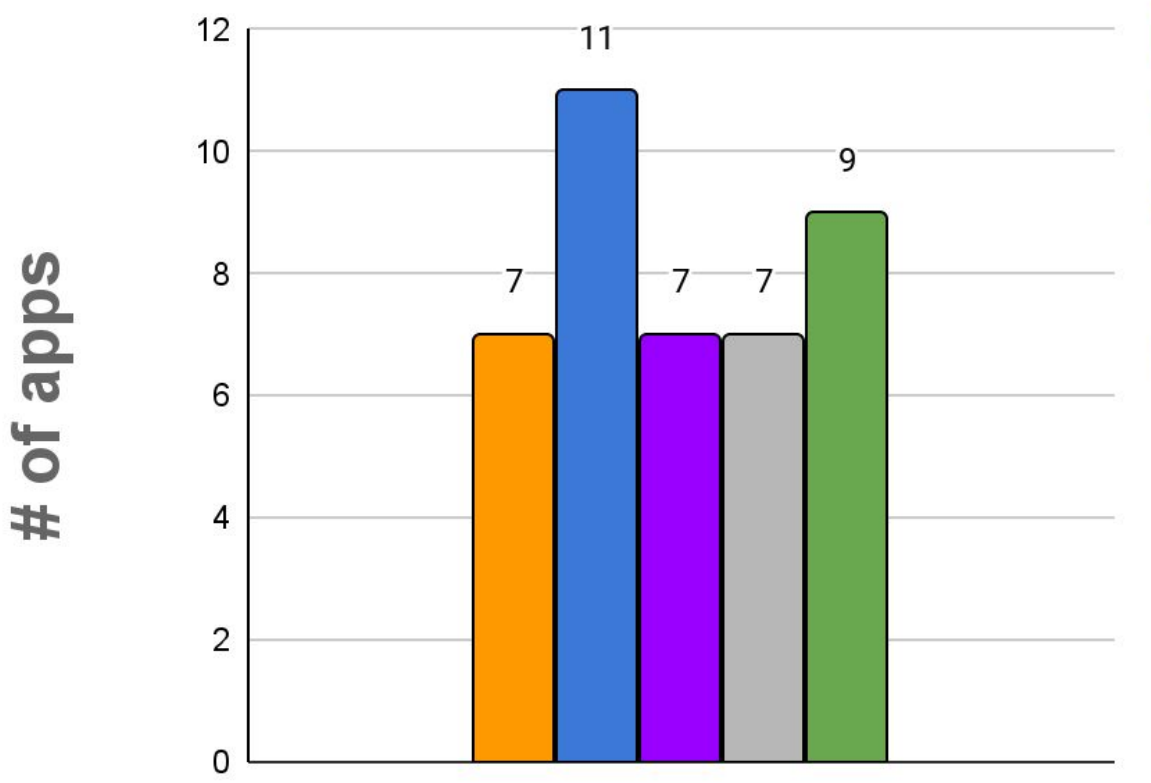
Key Findings



Created by Zohaib Bajwa
from Noun Project

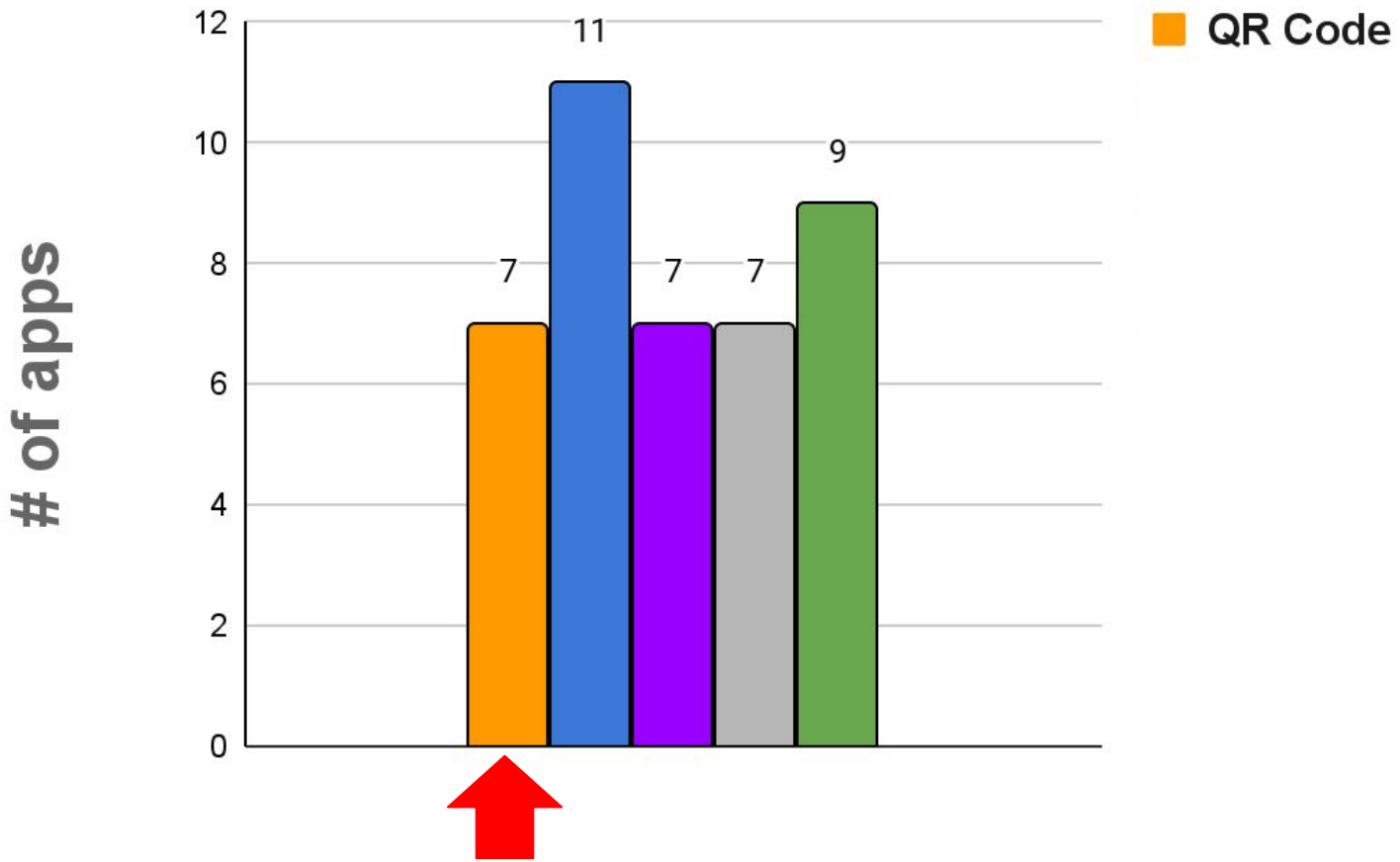
https://thenounproject.com/browse/icons/term/calendar

Backup Mechanisms



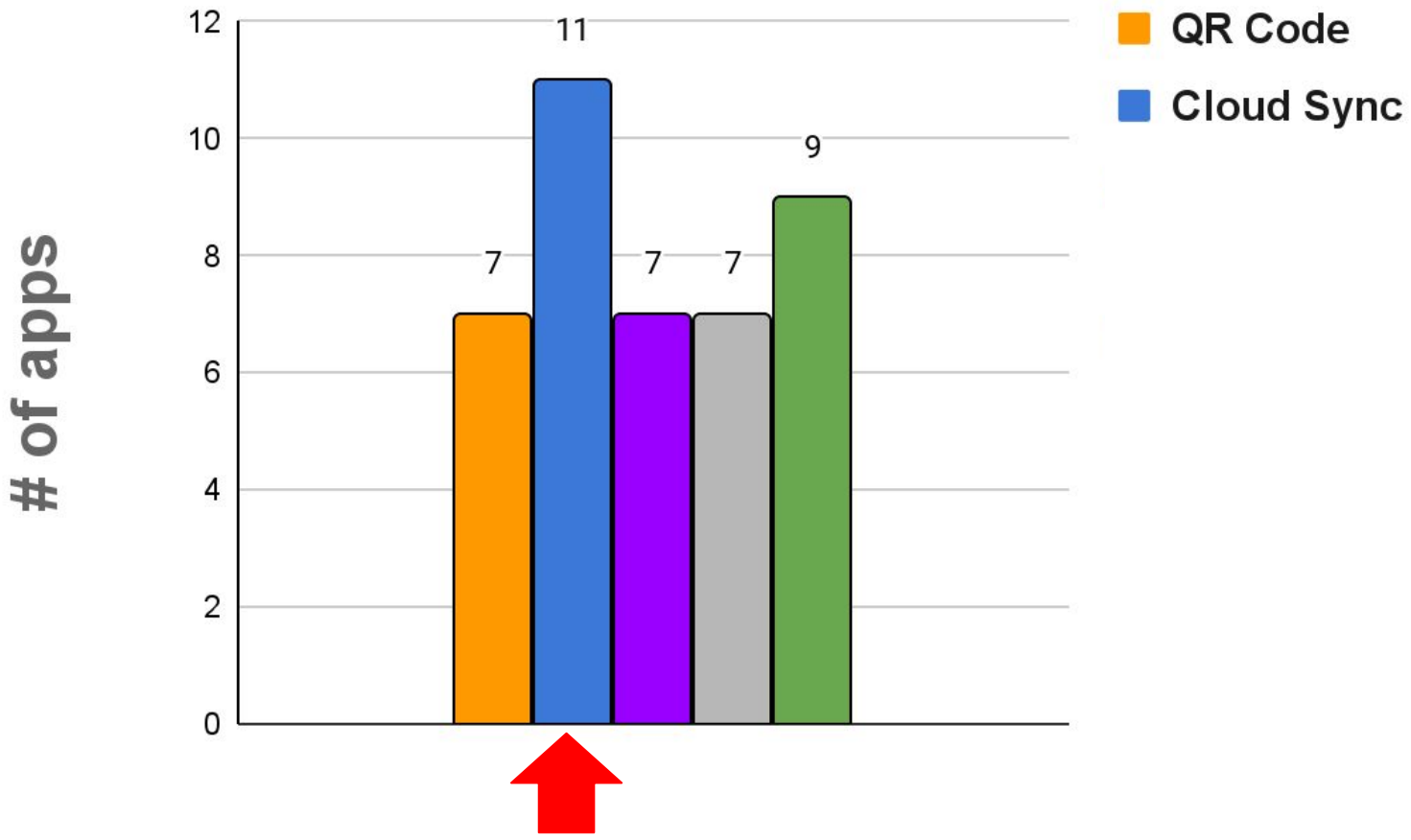
See paper
Table 1

Backup Mechanisms



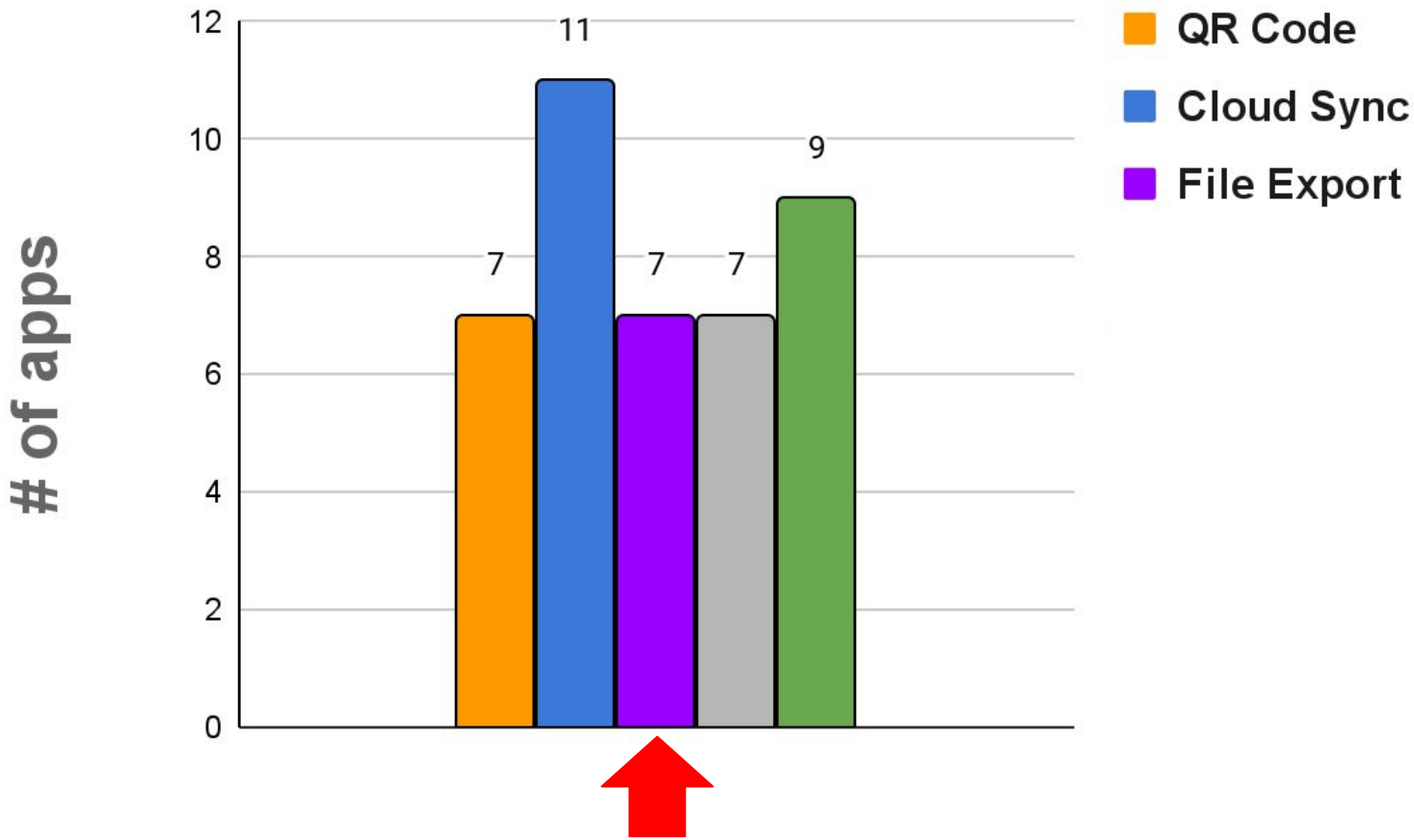
See paper Table 1

Backup Mechanisms



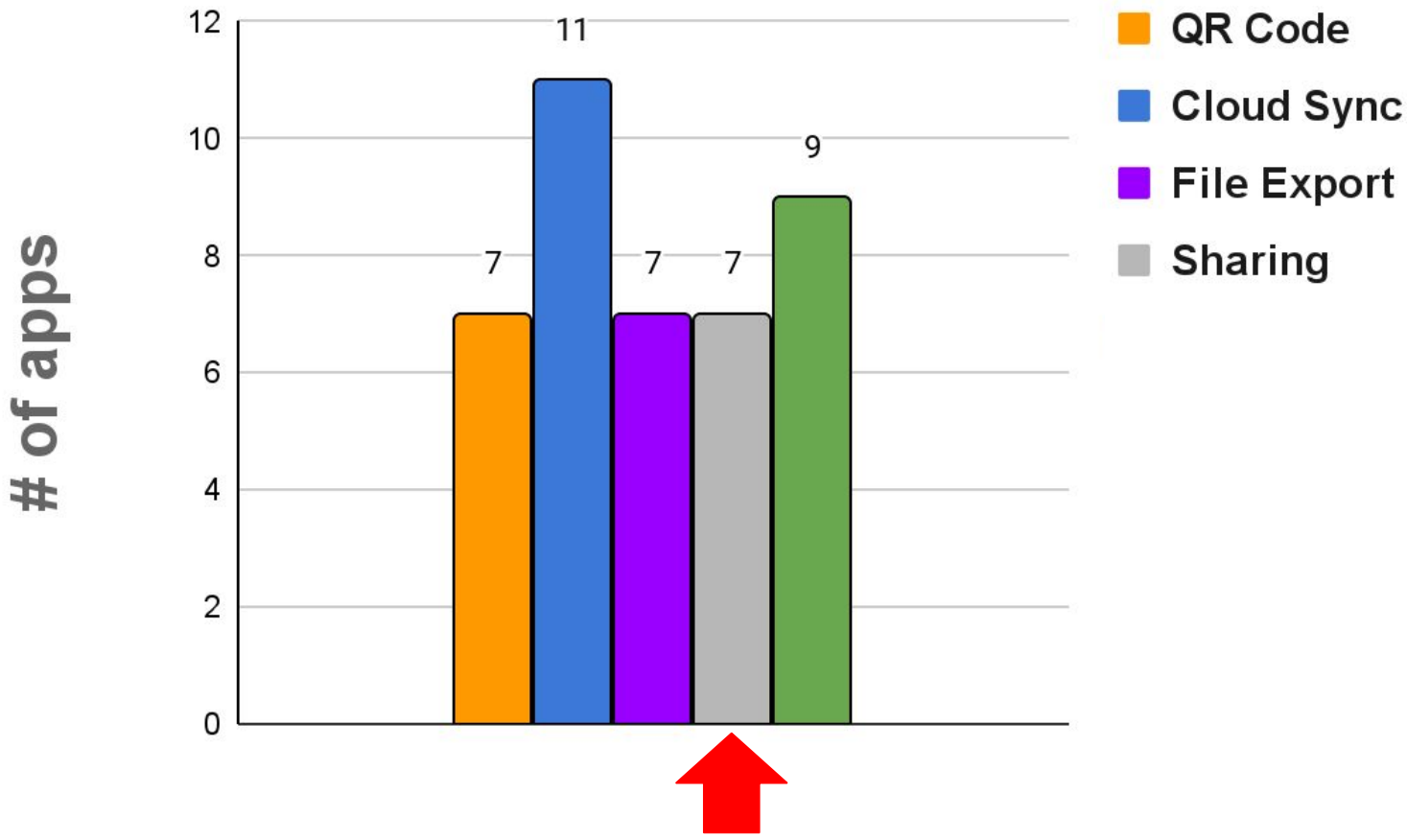
See paper Table 1

Backup Mechanisms



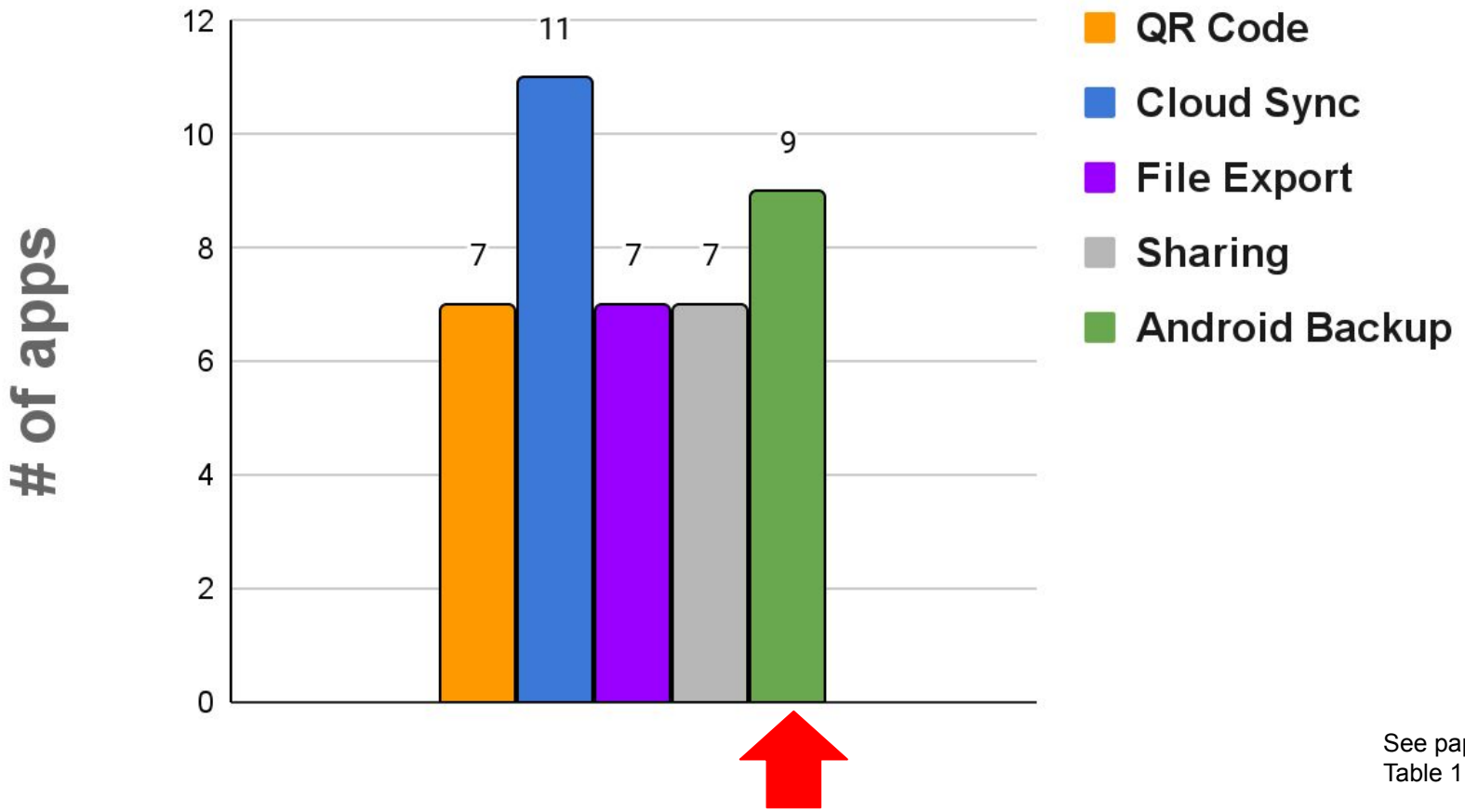
See paper Table 1

Backup Mechanisms



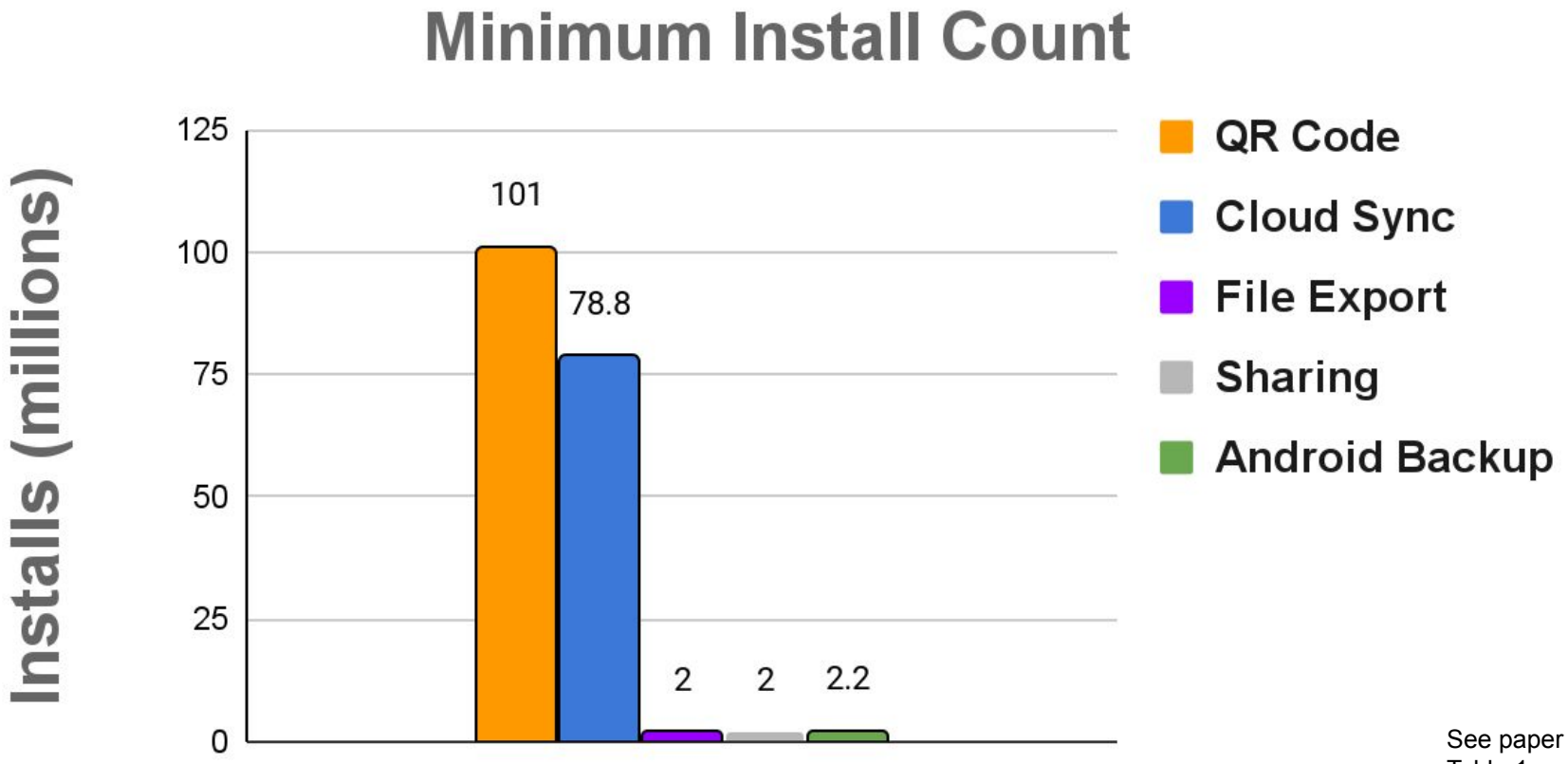
See paper Table 1

Backup Mechanisms



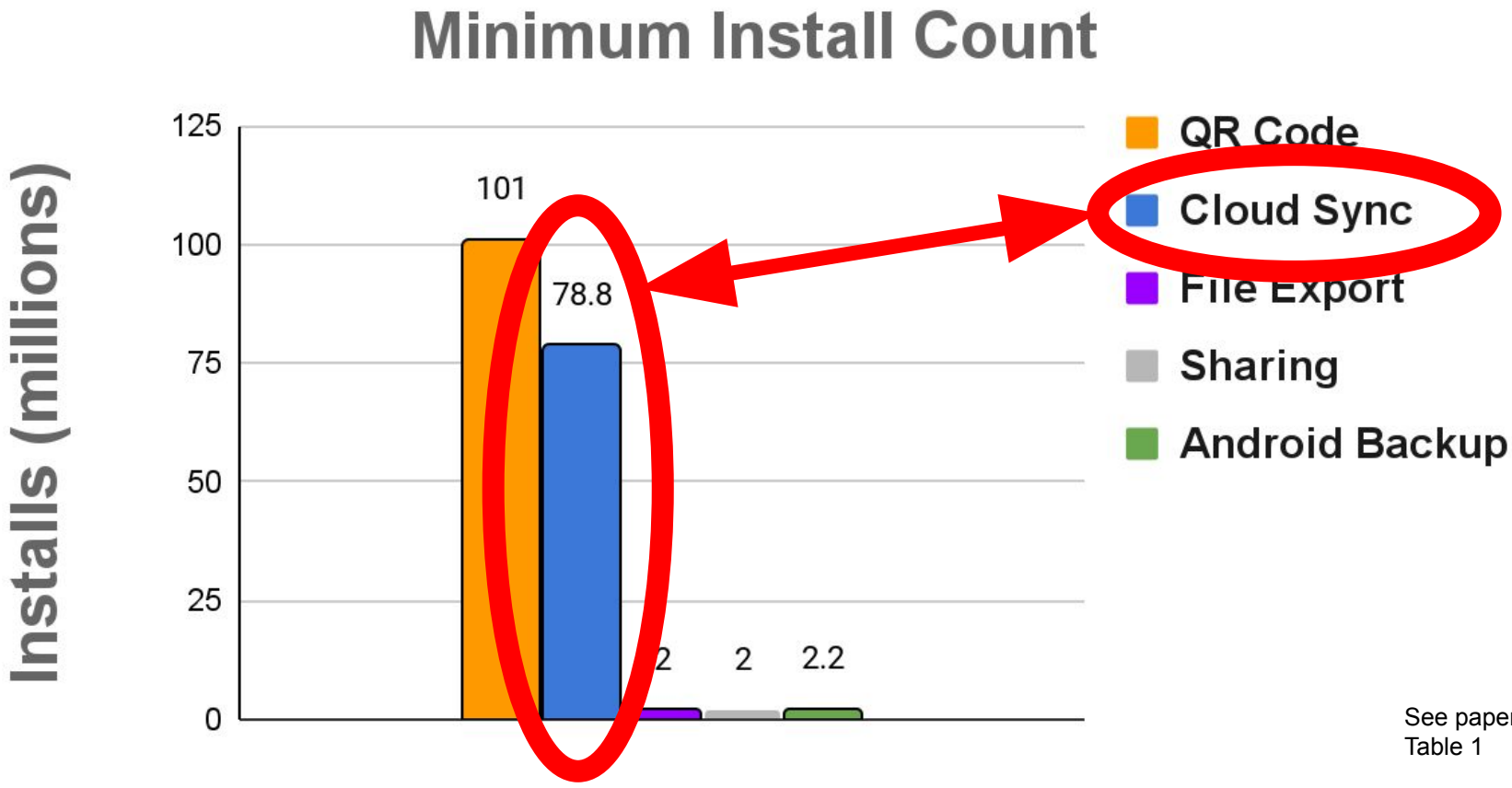
See paper Table 1

Backup Mechanisms



See paper Table 1

Backup Mechanisms



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

Authy 2-Factor Authentication
Authy

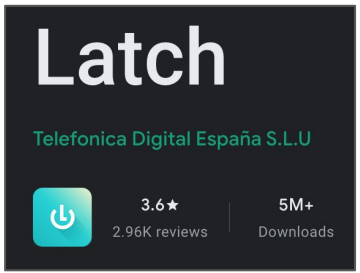
SAASPASS Authenticator 2FA App & Password Manager
SAASPASS

Salesforce Authenticator
Salesforce.com, inc.

Yandex.Key – your passwords
Yandex Apps

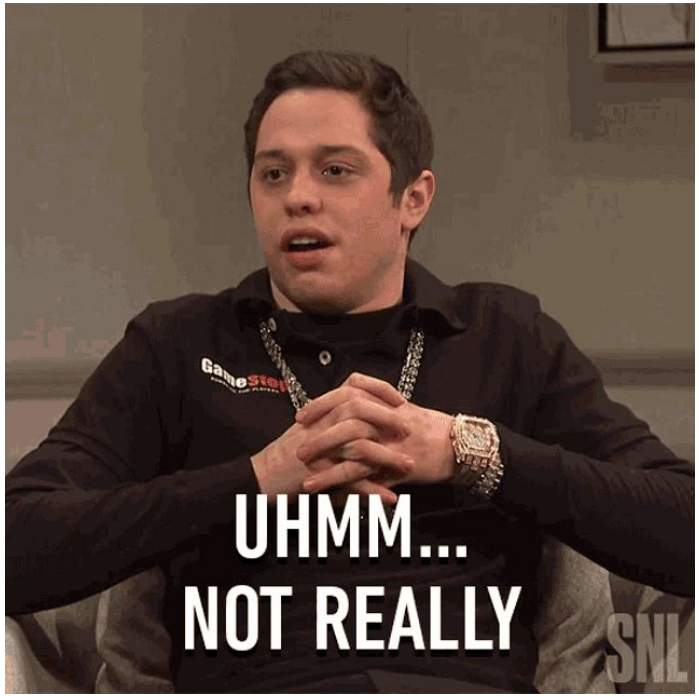
No Encryption

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

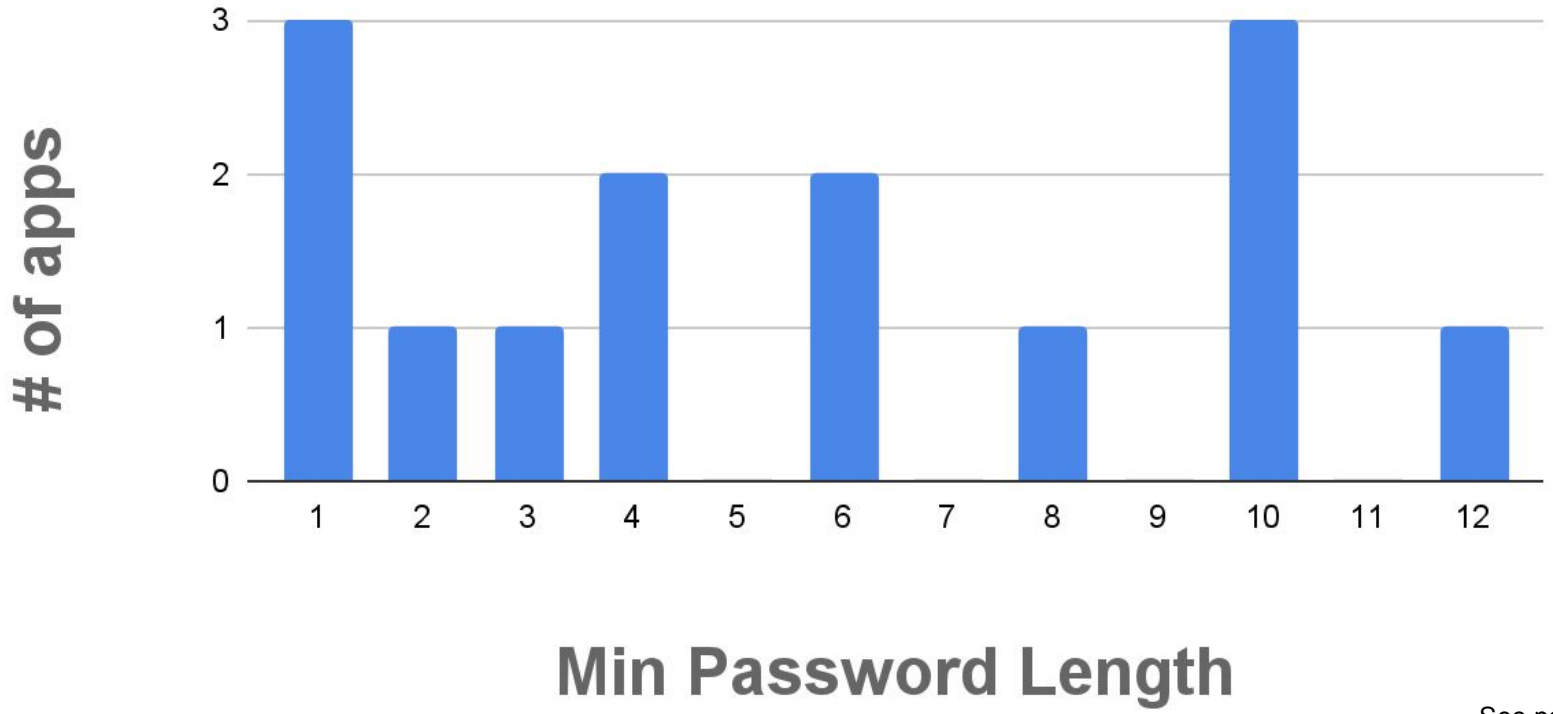
Microsoft Authenticator

Microsoft Corporation

50+ million installs

Weak Password = Weak Key

Severely Inadequate Password Policies

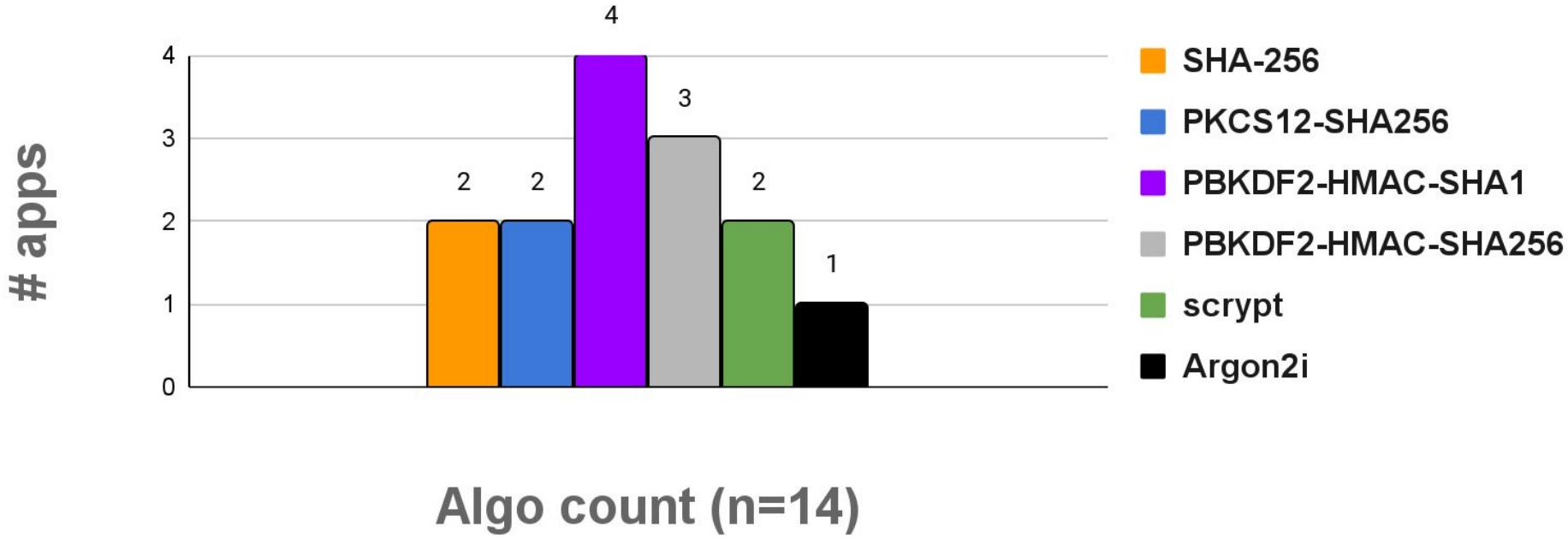


See paper Table 3

Severely Inadequate Password Policies

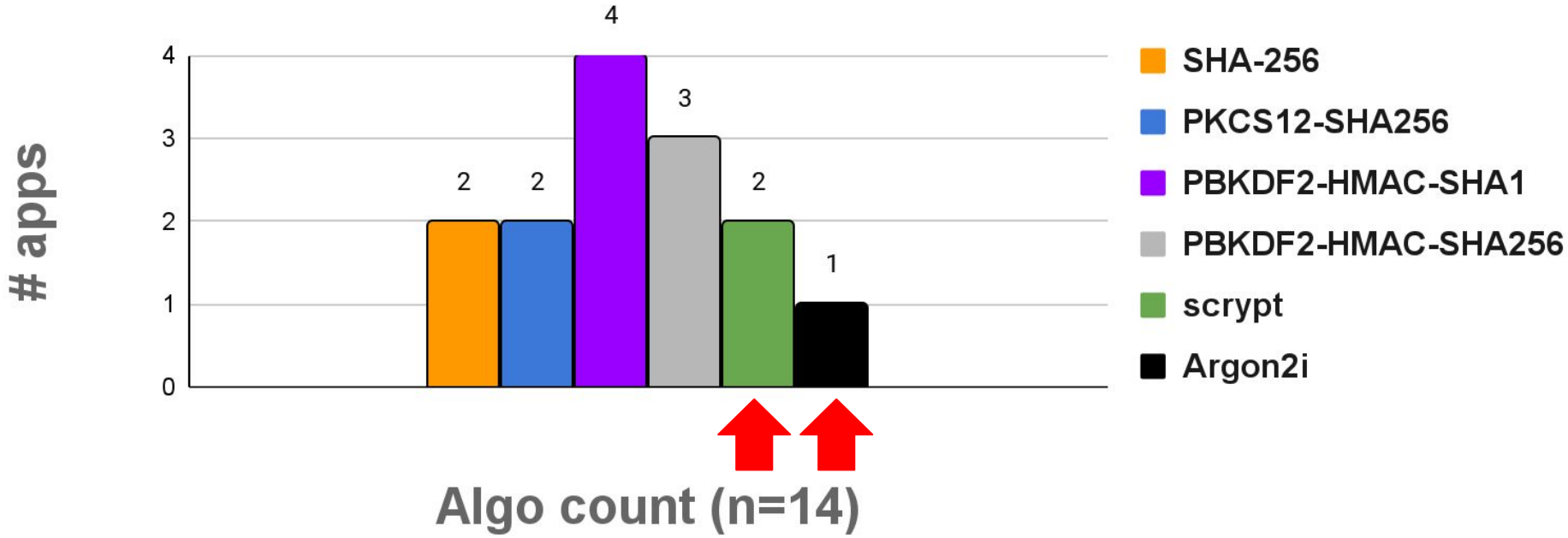
See paper Table 3

Weak Key Derivation



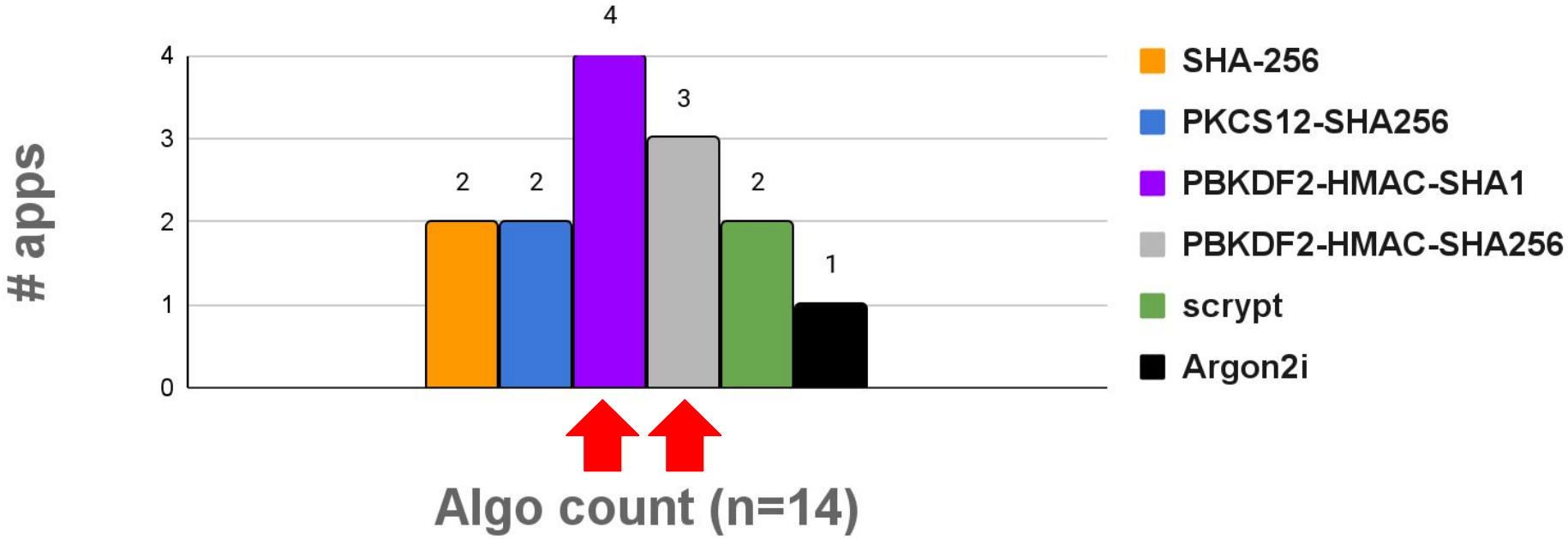
See paper Table 3

Weak Key Derivation



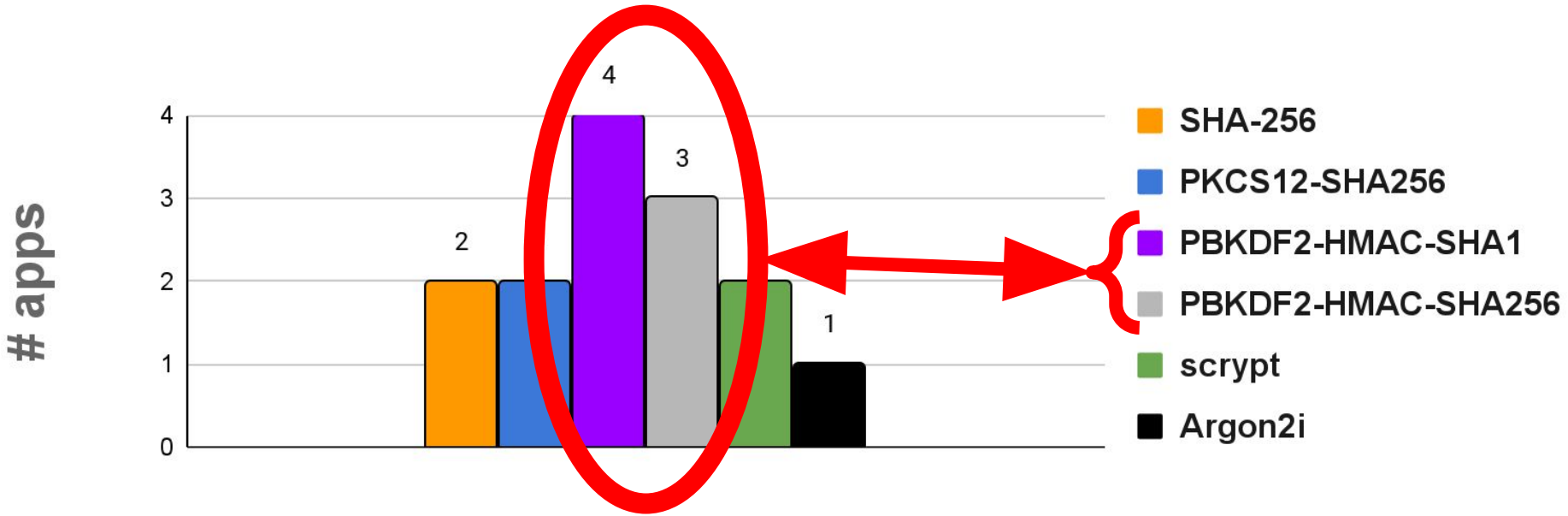
See paper Table 3

Weak Key Derivation



See paper Table 3

Weak Key Derivation



Weak PBKDF2 configurations

min = 10k, **median** = 10k, **max** = 160k

See paper Table 3

Where do keys go?

Poor Key Management

4 apps sent the ciphertext and key

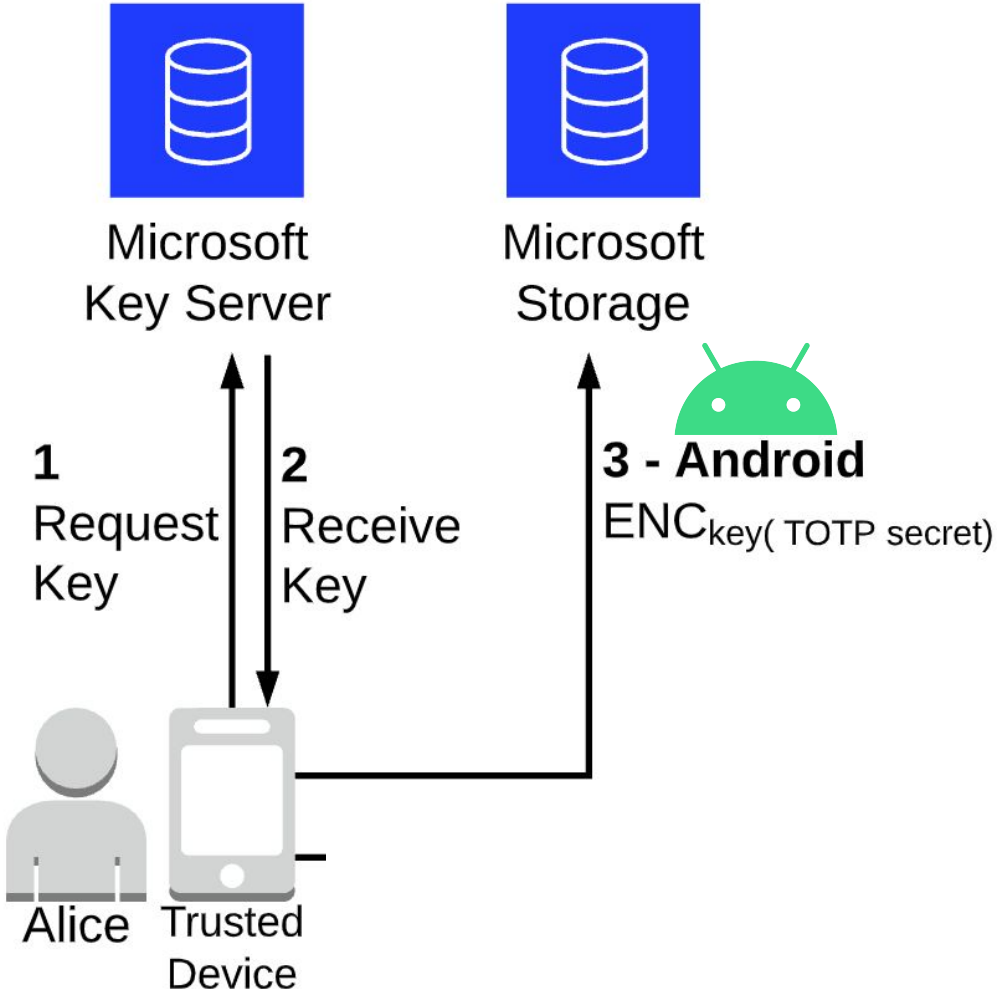
(or password from which it was derived)

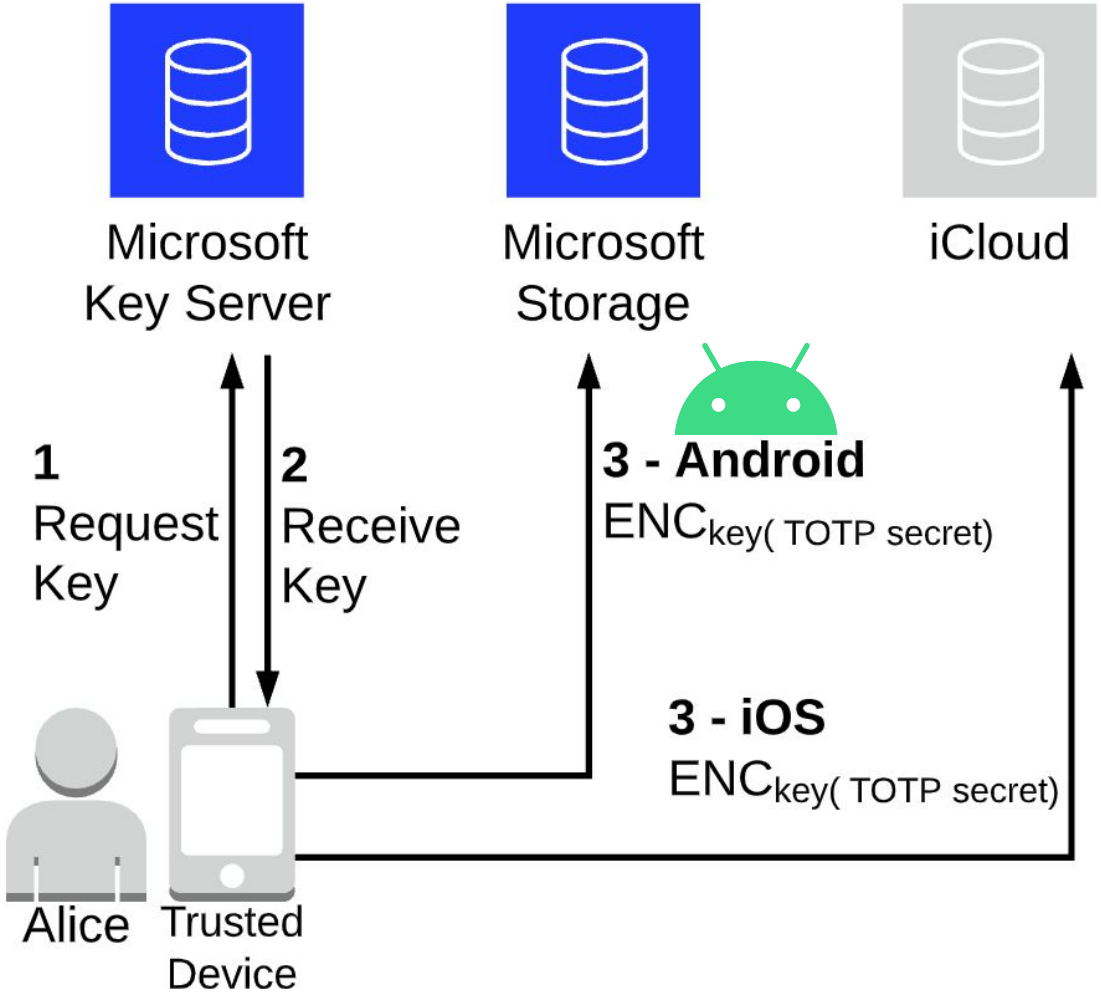
to the app developers

Microsoft Authenticator

Microsoft Corporation

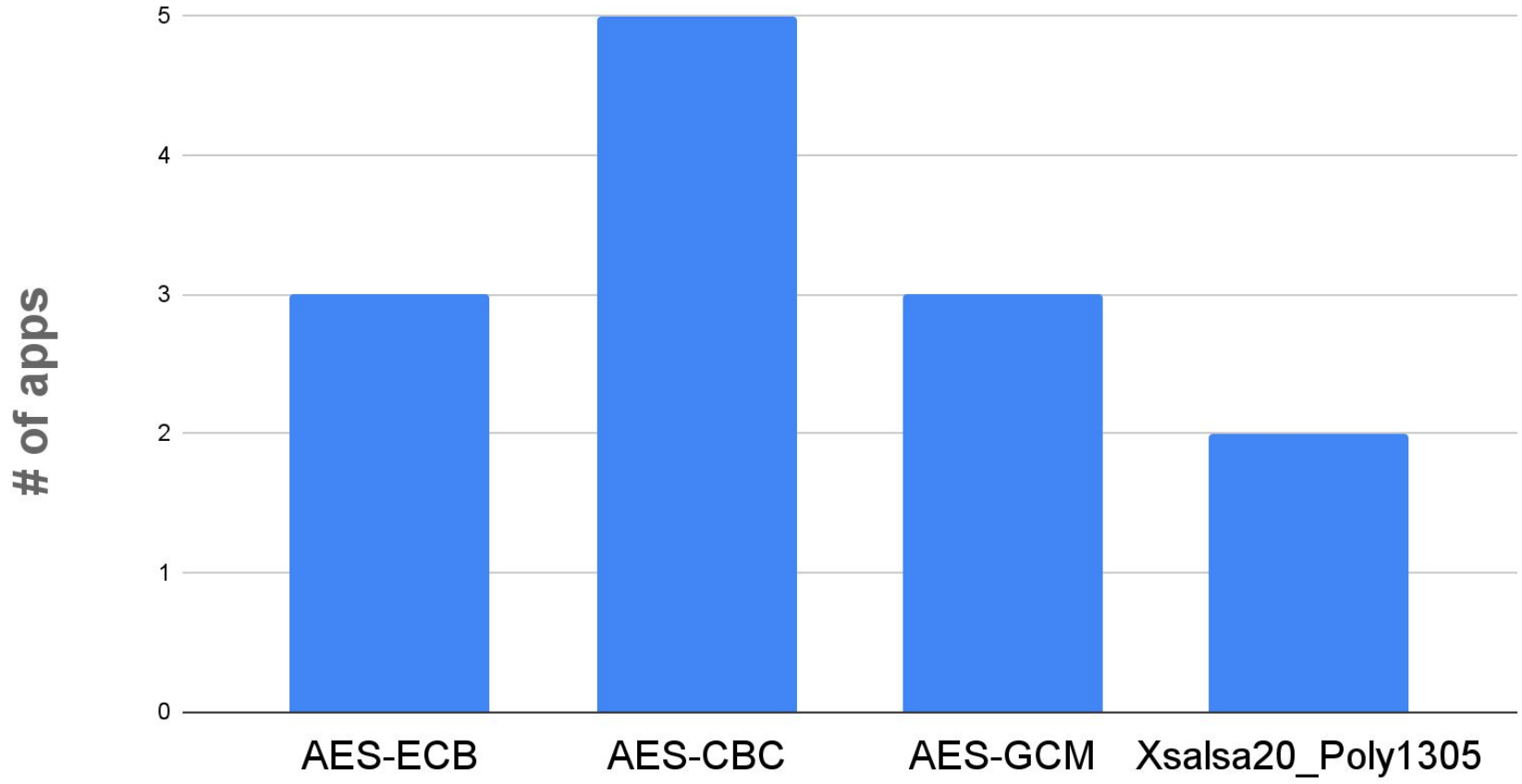
50+ million installs


Microsoft
Key Server
Microsoft
Storage
1
Request
Key
2
Receive
Key
3 - Android
ENC key(TOTP secret)
Alice
Trusted
Device

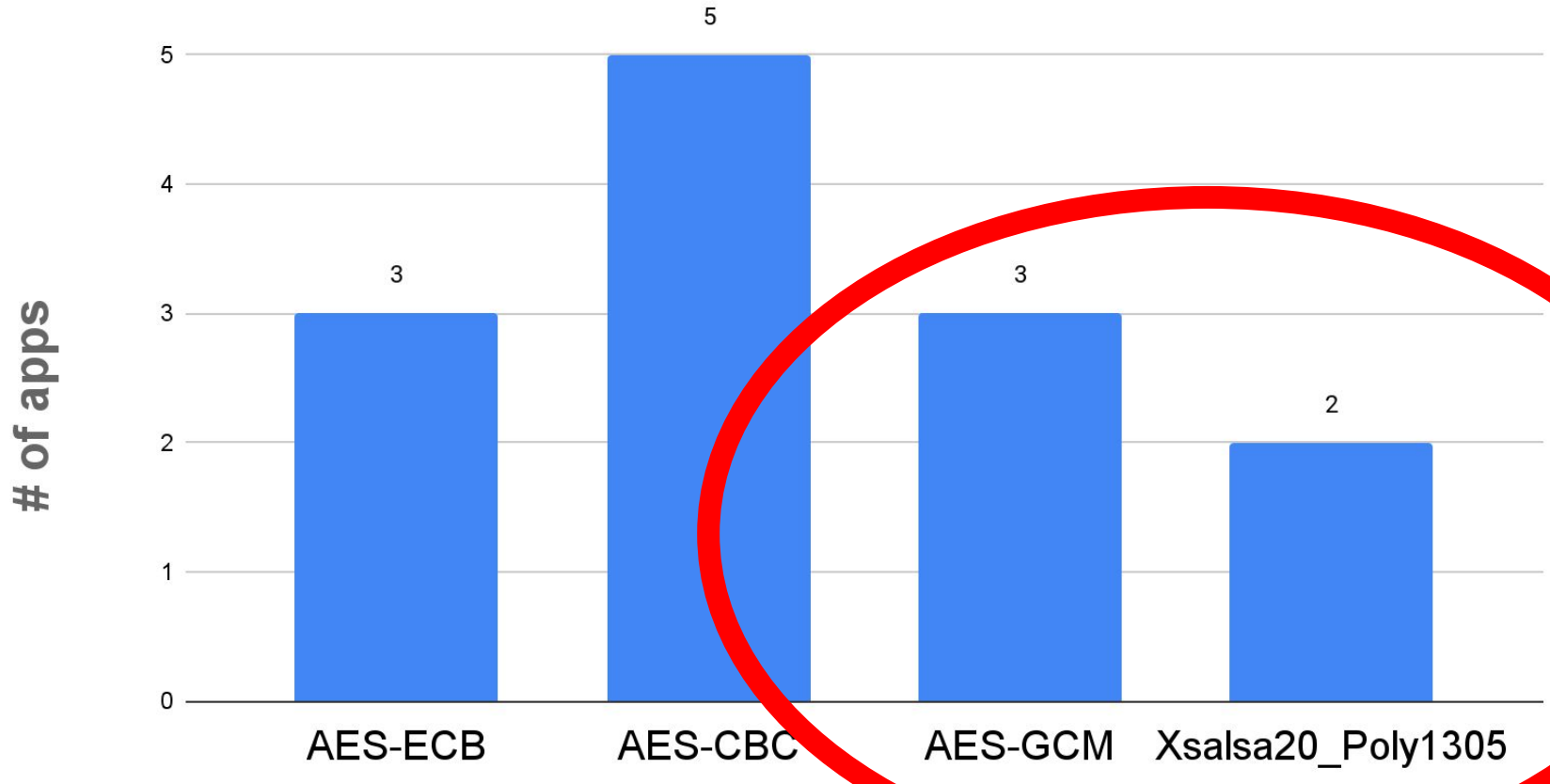

Microsoft
Key Server
Microsoft
Storage
iCloud
1
Request
Key
2
Receive
Key
3 - Android
ENCkey(TOTP secret)
3 - iOS
ENCkey(TOTP secret)
Alice
Trusted
Device

How are keys used?

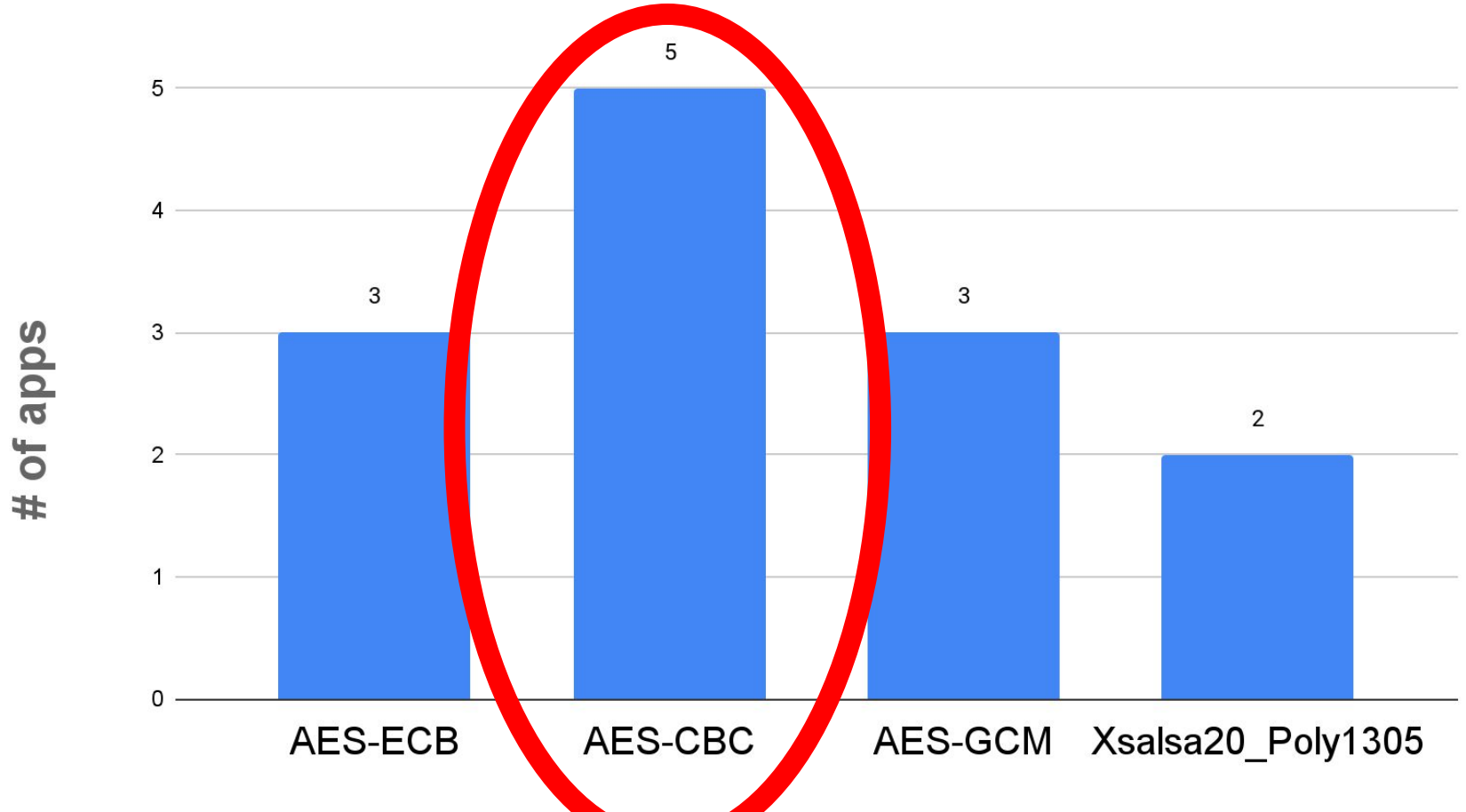
Encryption Algos



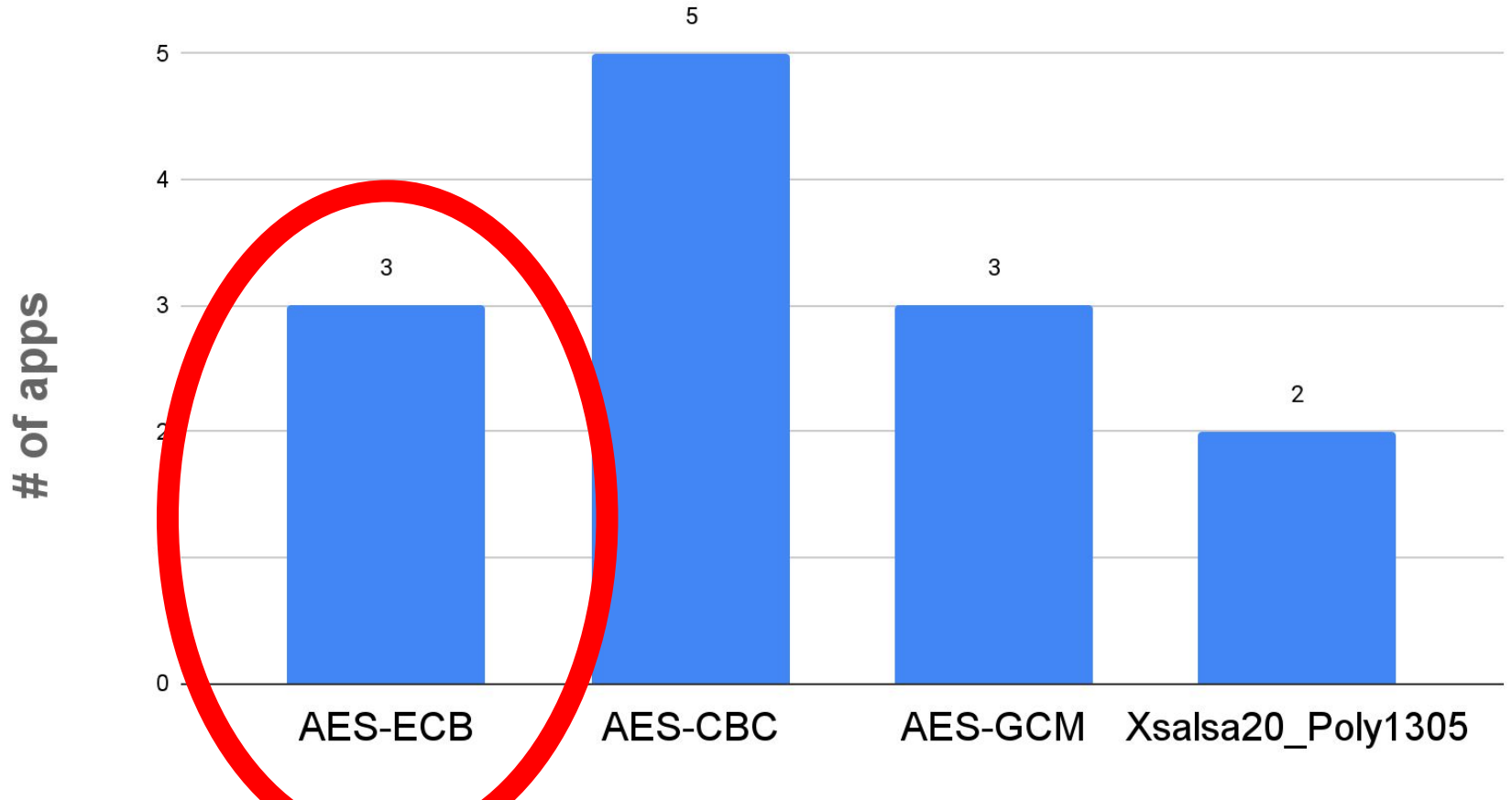
Encryption Algos



Encryption Algos



Encryption Algos



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

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

Abbreviated Name	APK id@version	Installs	Backup Mechanisms							
			QR Codes	Cloud Sync		File Export		Sharing		Android Backup
				Plaintext	Encrypted	Plaintext	Encrypted	Plaintext	Encrypted	
Google Authenticator	com.google.android.apps.authenticator2@v5.10	100M+	Y	-	-	-	-	-	-	-
Microsoft Authenticator	com.azure.authenticator@v6.2204.2757	50M+	-	-	Y*	-	-	-	-	-
Duo Mobile	com.duosecurity.duomobile@v4.15.0	10M+	-	-	Y	-	-	-	-	-
Twilio Authy	com.authy.authy@v24.8.5	10M+	-	-	Y	-	-	-	-	-
Latch	com.elevenpaths.android.latch@v2.2.4	5M+	-	Y	-	-	-	-	-	-
LastPass Authenticator	com.lastpass.authenticator@v2.5.0	1M+	-	-	(Y)	-	-	-	-	-
2FAS	com.twofasapp@v3.11.0	1M+	-	Y	Y*	Y	Y	Y	Y	-
Yandex.Key	ru.yandex.key@v2.7.0	1M+	-	-	Y*	-	-	-	-	-
FreeOTP Authenticator	org.fedorahosted.freeotp@v1.5	1M+	-	-	-	-	-	-	-	Y
Authenticator	com.pixplicity.auth@v1.0.6	500k+	Y	-	-	-	-	Y	Y*	-
Salesforce Authenticator	com.salesforce.authenticator@v3.8.5	500k+	-	-	Y*	-	-	-	-	-
Code Generator	net.codemonkey.otpgeneratorapp@v6.1	500k+	-	-	-	Y	-	-	-	Y
TOTP Authenticator	com.authenticator.authservice2@v1.89	100k+	Y	-	Y*	-	Y*	Y	Y	Y
Aegis Authenticator	com.beemdevelopment.aegis@v2.0.3	100k+	-	-	-	Y	Y	Y	Y	Y
Auth0 Guardian	com.auth0.guardian@v1.5.3	100k+	-	-	-	-	-	-	-	Y
App Authenticator	authentic.your.app.authenticator@v1.5	100k+	Y	-	-	-	Y*	Y	-	Y
andOTP	org.shadowice.flocke.andotp@v0.9.0.1-play	100k+	Y	-	-	Y	Y^	-	-	Y
Zoho OneAuth	com.zoho.accounts.oneauth@v2.1.0.5	100k+	-	-	Y*	-		-	-	-
Authenticator Pro	me.jmh.authenticatorpro@v1.15.10	100k+	-	-	-	Y	Y	-	-	-
SAASPASS	com.solidpass.saaspass@v2.2.28	100k+	-	Y	-	-	-	-	-	-
Authentic Password	authentic.password.authenticator.pro@v1.3	100k+	Y	-	-	-	-	Y	-	Y
Mobile Authenticator	authenticator.mobile.authenticator@v1.7	100k+	Y	-	-	-	-	Y	-	Y
	TOTAL apps	**-**	**7**	**3**	**9**	**5**	**6**	**7**	**4**	**9**
	TOTAL installs	**181.5M+**	**101M+**	**6.1M+**	**73.7M+**	**1.8M+**	**1.5M+**	**2M+**	**1.7M+**	**2.2M+**

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).

Abbreviated Name	Encrypted?	PII to use cloud backups					Backup Location	TOTP Data Leaked			Obtain Backup With...
		phone	email	name	dob	photo		secret	label	issuer	
Microsoft Authenticator	Yes*	Y	Y	Y	Y	-	activity.windows.com	Y	Y	Y	Microsoft account
Duo Mobile	Yes	-	Y	Y	-	Y	www.googleapis.com	-	Y^	Y^	Google account
Twilio Authy	Yes	Y	Y	-	-	-	api.authy.com	-	Y	Y	SMS only
Latch	No	-	Y	-	-	-	latch.elevenpaths.com	Y	Y	Y	Latch account
LastPass Authenticator	(Yes)	-	Y	-	-	-	(lastpass servers)	(Y)	(Y)	(Y)	Lastpass account
2FAS	No	-	Y	Y	-	Y	www.googleapis.com	Y	Y	Y	Google account
	Yes*	-	Y	Y	-	Y	www.googleapis.com	Y^	Y^	Y^	Google account
Yandex.Key	Yes*	Y	-	-	-	-	registrator.mobile.yandex.net	Y	Y	Y	SMS only
Salesforce Authenticator	Yes*	Y	-	-	-	-	authenticator-api.salesforce.com	Y	Y	Y	SMS only
TOTP Authenticator	Yes*	-	Y	Y	-	Y	www.googleapis.com	Y	Y	Y	Google account
Zoho OneAuth	Yes*	-	Y	-	-	-	accounts.zoho.com	Y	Y	Y	Zoho account
SAASPASS	No	Y	-	-	-	-	104.154.49.147	Y	Y	Y	SMS only

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).

Abbreviated Name	Key Source	Password Min Len	KDF and Configuration	KDF Salt	Encryption Algorithm	Ciphertext Integrity	Decryption Heuristic
Microsoft Authenticator	Random*	n/a	n/a	n/a	AES-128-CBC	HMAC-SHA256	n/a
Zoho OneAuth	Password*	3	SHA-256 i = 1	none	AES-256-ECB	none	Base32
Salesforce Authenticator	Password*	4	PBKDF2-HMAC-SHA256 i = 10,000	random	AES-256-CBC	none	JSON
Yandex.Key	Password*	6	scrypt N = 2^15, r = 20, p = 1	random	Xsalsa20_Poly1305	AEAD	n/a
TOTP Authenticator	Password	8	SHA-256 i = 1	none	AES-256-CBC	none	JSON
Authenticator	Password	10	PKCS12-SHA256 i = 65,536	hard coded	AES-256-ECB	none	URI
App Authenticator	Password	10	PKCS12-SHA256 i = 65,536	hard coded	AES-256-ECB	none	URI
Auth0 Guardian	Password	1	(PBKDF2-HMAC-SHA1) (i = 10,000)	(random)	(AES-256)	(HMAC)	(n/a)
Authenticator Pro	Password	1	PBKDF2-HMAC-SHA1 i = 64,000	random	AES-256-CBC	none	JSON
2FAS	Password OpenPGP	1	PBKDF2-HMAC-SHA256 i = 10,000	random	AES-256-GCM	AEAD	n/a
Aegis Authenticator	Password	2	scrypt N = 2^15, r = 8, p = 1	random	AES-256-GCM	AEAD	n/a
andOTP	Password	4	PBKDF2-HMAC-SHA1 i = [140,000 - 160,000]	random	AES-256-GCM	AEAD	n/a
Twilio Authy	Password	6	PBKDF2-HMAC-SHA1 i = 10,000	random	AES-256-CBC	none	Base32
Duo Mobile	Password	10	argon2i m = 128 Mb, t = 6, p = 1	random	Xsalsa20_Poly1305	AEAD	n/a
LastPass Authenticator	Password	12	(PBKDF2-HMAC-SHA256) (i = 100,100)	(random)	(AES-256)	(HMAC)	(n/a)

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).