CSE509 Computer System Security

Stony Brook University

2023-03-07

Authentication

Michalis Polychronakis

Stony Brook University

Authentication

The process of verifying someone's identity or role

- User, device, service, request, …

What is identity?

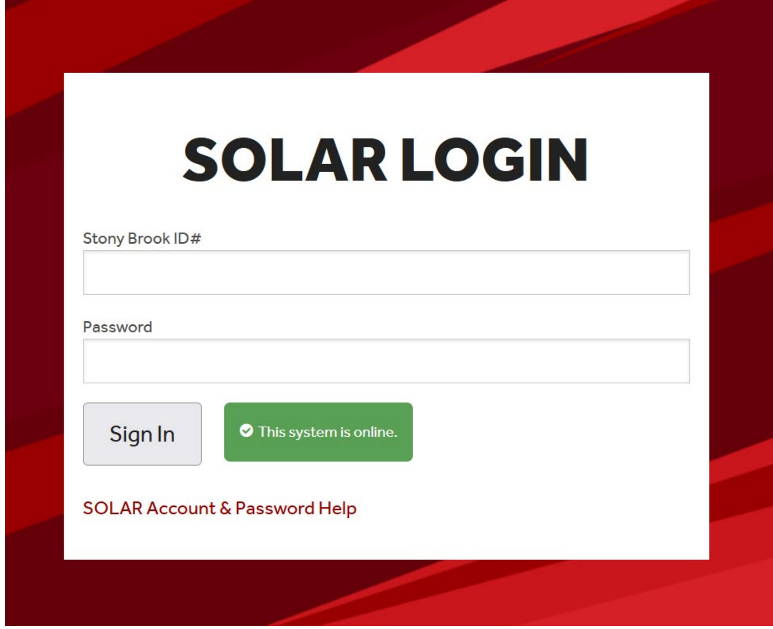
- Which characteristics uniquely identify an entity?

Authentication is a critical service

- Enables communicating parties to verify the identity of their peers
- Many other security mechanisms rely on it

Two main types

- Human to computer
- Computer to computer



SOLAR LOGIN

Stony Brook ID#

Password

Credentials

Evidence used to prove an identity

User Authentication: credentials supplied by a person

- *Something you know*
- *Something you have*
- *Something you are*

Computer authentication: crypto, location

- Computers (in contrast to humans) can “remember” large secrets (keys) and perform complex cryptographic operations
- Location: evidence that an entity is at a specific place (IP, subnet, switch port, …)

Authentication can be delegated

- The verifying entity accepts that a trusted third party has already established authentication

Something You Know: Password-based Authentication

Passwords, passphrases, pins, key-phrases, access codes, …

Good passwords are easy to remember and hard to guess

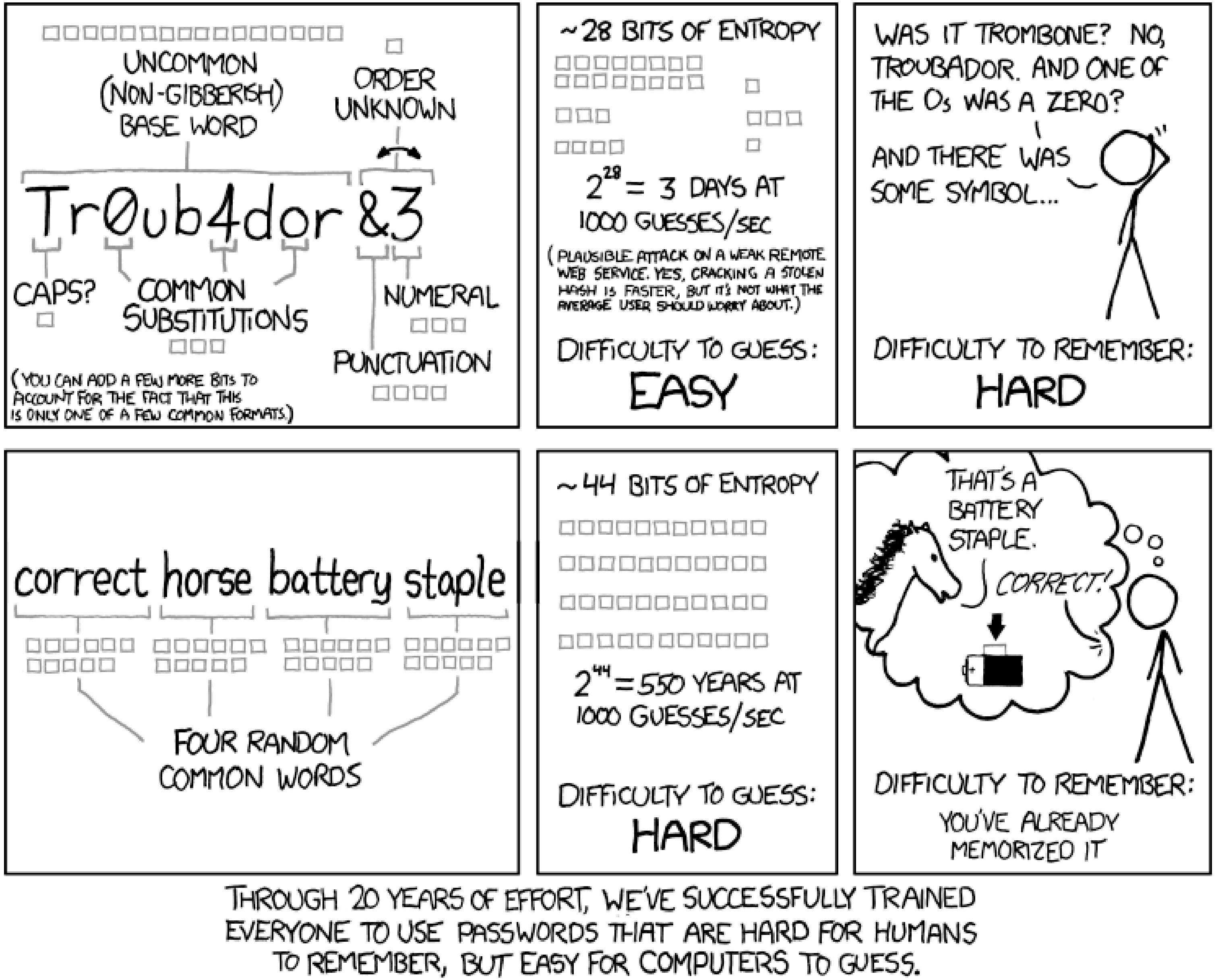
- Easy to remember ➔ easy to guess
- Hard to guess ➔ hard to remember
- Bad ideas: date of birth, SSN, zip code, favorite team name, …

Password space (bits) depends on:

- Password length
- Character set

Better way to think about strong passwords: **long passphrases**

- Can be combined with custom variations, symbols, numbers, capitalization, …


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS.)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

© XKCD - https://xkcd.com/936/

Password Policies (often have the opposite effect)

Password rules (often miss the point)

"At least one special character," "Minimum/Maximum length of 8/12 characters," "Must contain at least one number," "Must contain at least one capital letter"

Makes passwords hard to remember! ➔ encourages password reuse

Better: encourage long passphrases, evaluate strength on-the-fly

Periodic password changing (does more harm than good)

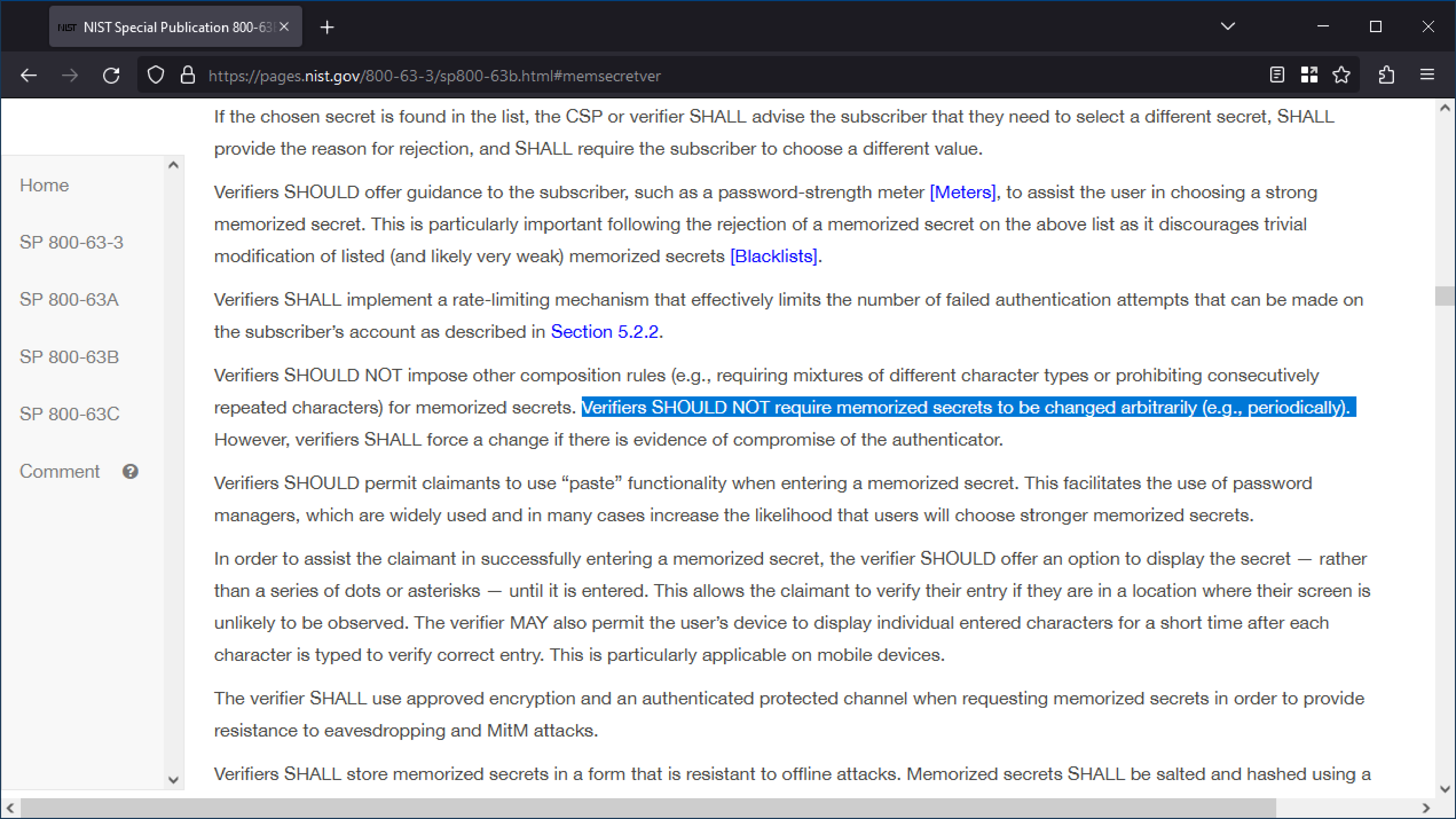
"You haven't changed your password in the last 90 days"

Probably too late anyway if password has already been stolen

Makes remembering passwords harder ➔ more password resets

Hinders the use of password managers (!)

What users do: `password1` ➔ `password2` ➔ `password3` ➔ …


NIST Special Publication 800-63B
https://pages.nist.gov/800-63-3/sp800-63b.html#memsecretver
Home
SP 800-63-3
SP 800-63A
SP 800-63B
SP 800-63C
Comment

If the chosen secret is found in the list, the CSP or verifier SHALL advise the subscriber that they need to select a different secret, SHALL provide the reason for rejection, and SHALL require the subscriber to choose a different value.

Verifiers SHOULD offer guidance to the subscriber, such as a password-strength meter [Meters], to assist the user in choosing a strong memorized secret. This is particularly important following the rejection of a memorized secret on the above list as it discourages trivial modification of listed (and likely very weak) memorized secrets [Blacklists].

Verifiers SHALL implement a rate-limiting mechanism that effectively limits the number of failed authentication attempts that can be made on the subscriber's account as described in Section 5.2.2.

Verifiers SHOULD NOT impose other composition rules (e.g., requiring mixtures of different character types or prohibiting consecutively repeated characters) for memorized secrets. Verifiers SHOULD NOT require memorized secrets to be changed arbitrarily (e.g., periodically). However, verifiers SHALL force a change if there is evidence of compromise of the authenticator.

Verifiers SHOULD permit claimants to use "paste" functionality when entering a memorized secret. This facilitates the use of password managers, which are widely used and in many cases increase the likelihood that users will choose stronger memorized secrets.

In order to assist the claimant in successfully entering a memorized secret, the verifier SHOULD offer an option to display the secret — rather than a series of dots or asterisks — until it is entered. This allows the claimant to verify their entry if they are in a location where their screen is unlikely to be observed. The verifier MAY also permit the user's device to display individual entered characters for a short time after each character is typed to verify correct entry. This is particularly applicable on mobile devices.

The verifier SHALL use approved encryption and an authenticated protected channel when requesting memorized secrets in order to provide resistance to eavesdropping and MitM attacks.

Verifiers SHALL store memorized secrets in a form that is resistant to offline attacks. Memorized secrets SHALL be salted and hashed using a

Attacking Passwords

Offline cracking

Online guessing

Brute force attacks

Eavesdropping

Capturing

Password Storage

Storing passwords as plaintext is disastrous

- Better way: store a cryptographic hash of the password

Even better: store the hash of a “salted” version of the password

- Defend against *dictionary attacks*: prevent precomputation of hash values (wordlists of popular passwords, rainbow tables, …)
- Even if two users happen to have the same password, their hash values will be different ➔ need to be cracked separately
- Salting *does not* make brute-force guessing a given password harder!

Username	Salt	Password hash
Bobbie	4238	h(4238, $uperman)
Tony	2918	h(2918, 63%TaeFF)
Mitsos	6902	h(6902, zour1da)
Mark	1694	h(1694, Rockybrook#1)

Password databases are still getting leaked…

Password Cracking

Exhaustive search ➔ infeasible for large password spaces

Dictionary attacks (words, real user passwords from previous leaks, …)

Variations, common patterns, structure rules

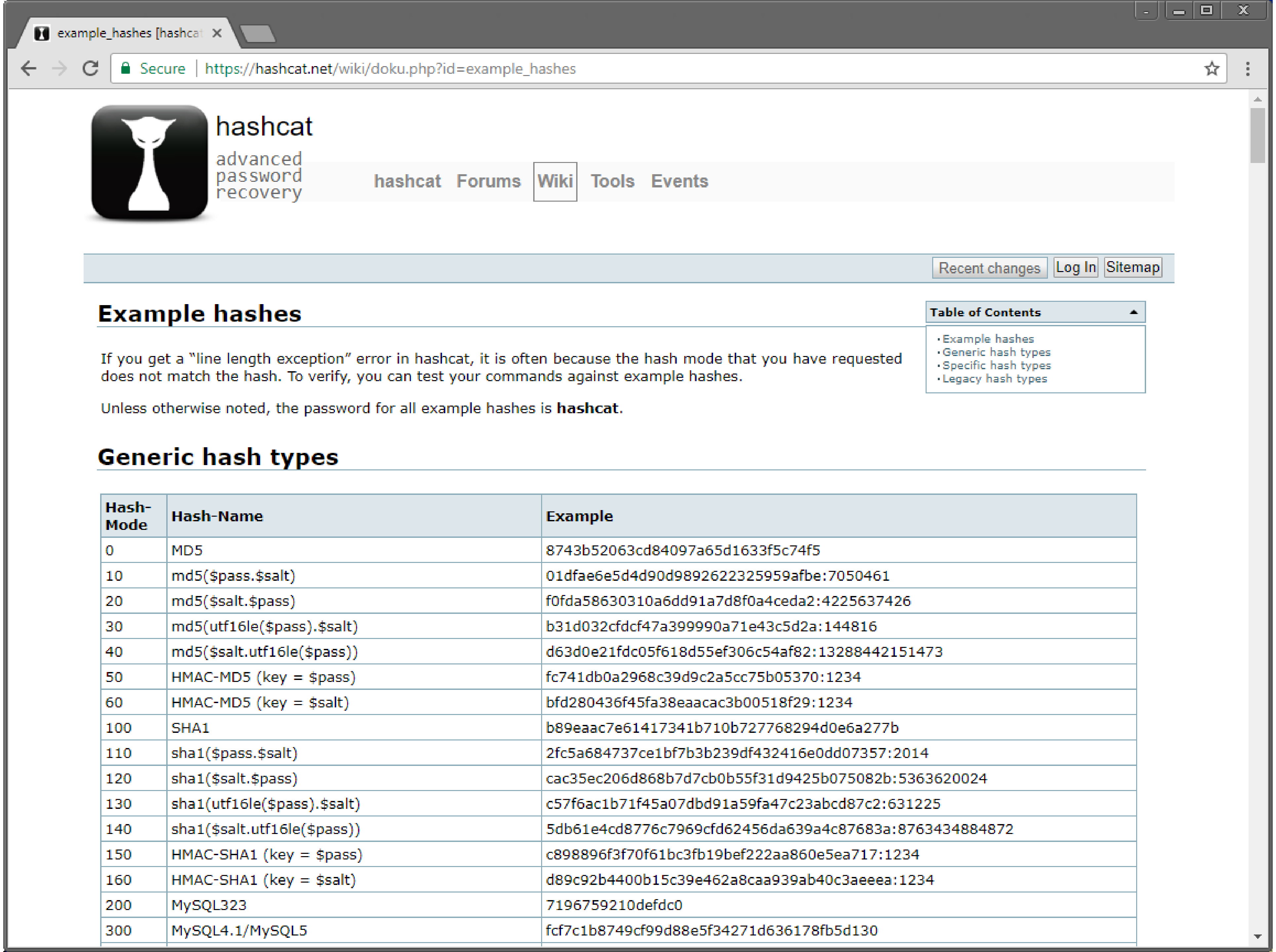
- Prepend/append symbols/numbers/dates, weird capitalization, l33tspeak, visually similar characters, intended misspellings, …

Target-specific information

- DOB, family names, favorite team, pets, hobbies, anniversaries, language, slang, …
- Easy to acquire from social networking services and other public sites
- **Particularly effective against "security questions"**

Advanced techniques

- Probabilistic context-free grammars, Markov models, …



Example hashes

Table of Contents

If you get a "line length exception" error in hashcat, it is often because the hash mode that you have requested does not match the hash. To verify, you can test your commands against example hashes.

Unless otherwise noted, the password for all example hashes is **hashcat**.

Generic hash types

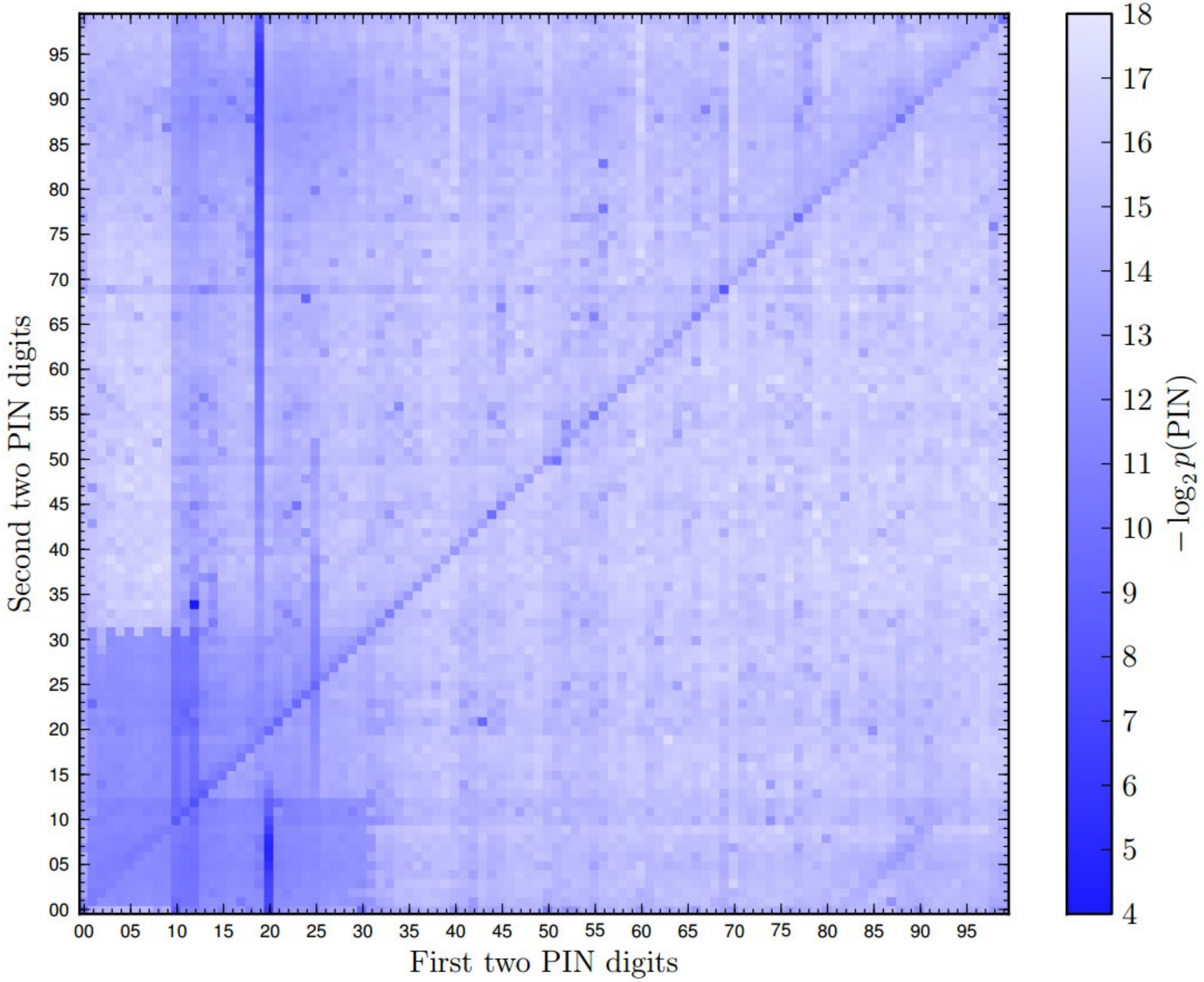
Hash-Mode	Hash-Name	Example
0	MD5	8743b52063cd84097a65d1633f5c74f5
10	md5($pass.$salt)	01dfae6e5d4d90d9892622325959afbe:7050461
20	md5($salt.$pass)	f0fda58630310a6dd91a7d8f0a4ceda2:4225637426
30	md5(utf16le($pass).$salt)	b31d032cfdcf47a399990a71e43c5d2a:144816
40	md5($salt.utf16le($pass))	d63d0e21fdc05f618d55ef306c54af82:13288442151473
50	HMAC-MD5 (key = $pass)	fc741db0a2968c39d9c2a5cc75b05370:1234
60	HMAC-MD5 (key = $salt)	bfd280436f45fa38eaacac3b00518f29:1234
100	SHA1	b89eaac7e61417341b710b727768294d0e6a277b
110	sha1($pass.$salt)	2fc5a684737ce1bf7b3b239df432416e0dd07357:2014
120	sha1($salt.$pass)	cac35ec206d868b7d7cb0b55f31d9425b075082b:5363620024
130	sha1(utf16le($pass).$salt)	c57f6ac1b71f45a07dbd91a59fa47c23abcd87c2:631225
140	sha1($salt.utf16le($pass))	5db61e4cd8776c7969cfd62456da639a4c87683a:8763434884872
150	HMAC-SHA1 (key = $pass)	c898896f3f70f61bc3fb19bef222aa860e5ea717:1234
160	HMAC-SHA1 (key = $salt)	d89c92b4400b15c39e462a8caa939ab40c3aeeea:1234
200	MySQL323	7196759210defdc0
300	MySQL4.1/MySQL5	fcf7c1b8749cf99d88e5f34271d636178fb5d130

50 Most-used (Worse) Passwords

- 123456
- 123456789
- picture1
- password
- 12345678
- 111111
- 123123
- 12345
- 1234567890
- senha
- 1234567
- qwerty
- abc123
- Million2
- 000000
- 1234
- iloveyou
- aaron431
- password1
- qqww1122
- 123
- omgpop
- 123321
- 654321
- qwertyuiop
- qwer123456
- 123456a
- a123456
- 666666
- asdfghjkl
- ashley
- 987654321
- unknown
- zxcvbnm
- 112233
- chatbooks
- 20100728
- 123123123
- princess
- jacket025
- evite
- 123abc
- 123qwe
- sunshine
- 121212
- dragon
- 1q2w3e4r
- 5201314
- 159753
- 0123456789

https://en.wikipedia.org/wiki/List_of_the_most_common_passwords

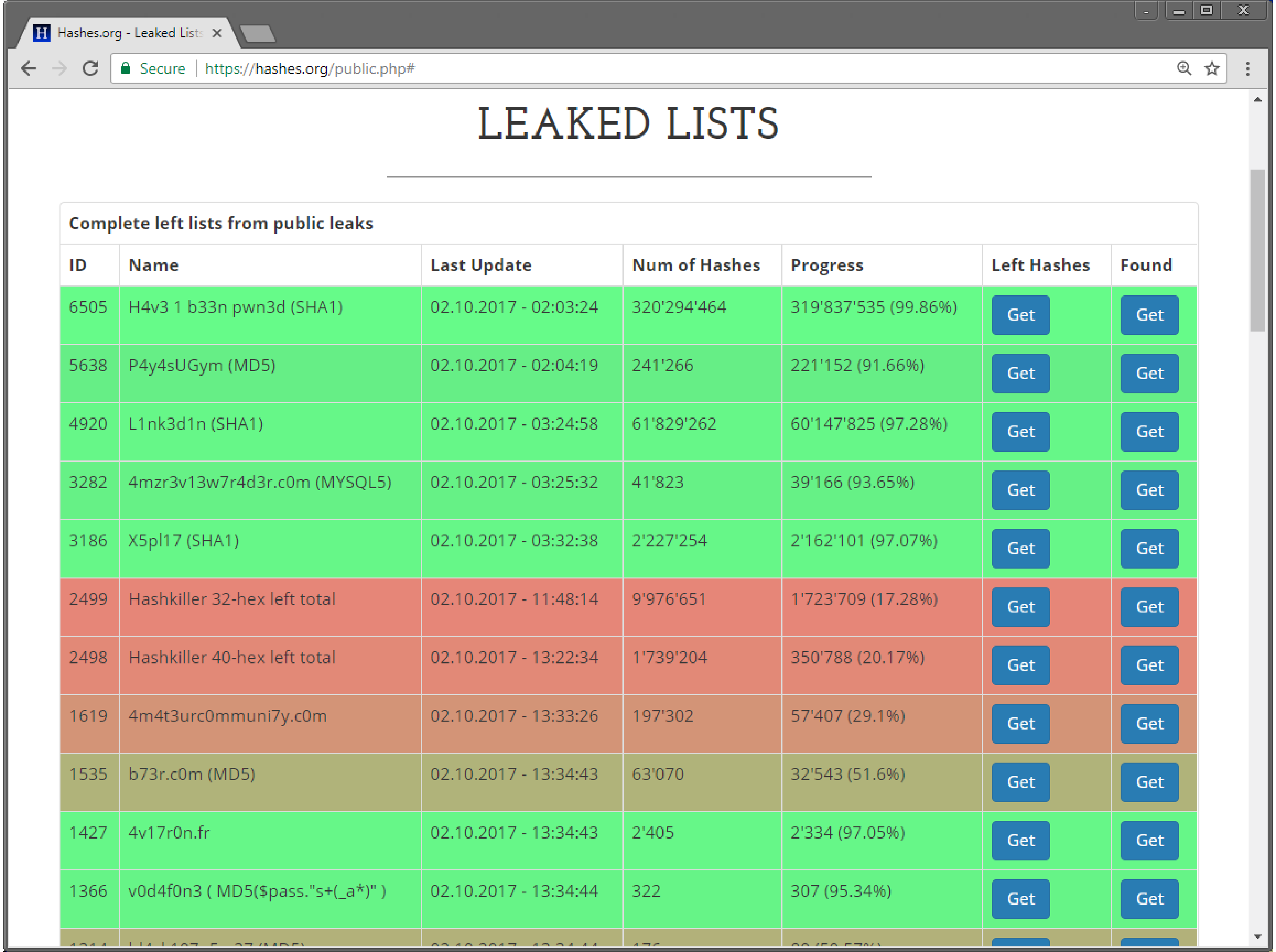
Distribution of 4-digit sequences within RockYou passwords



A birthday present every eleven wallets? The security of customer-chosen banking PINs. Joseph Bonneau, Sören Preibusch and Ross Anderson – FC '12

Wordlists

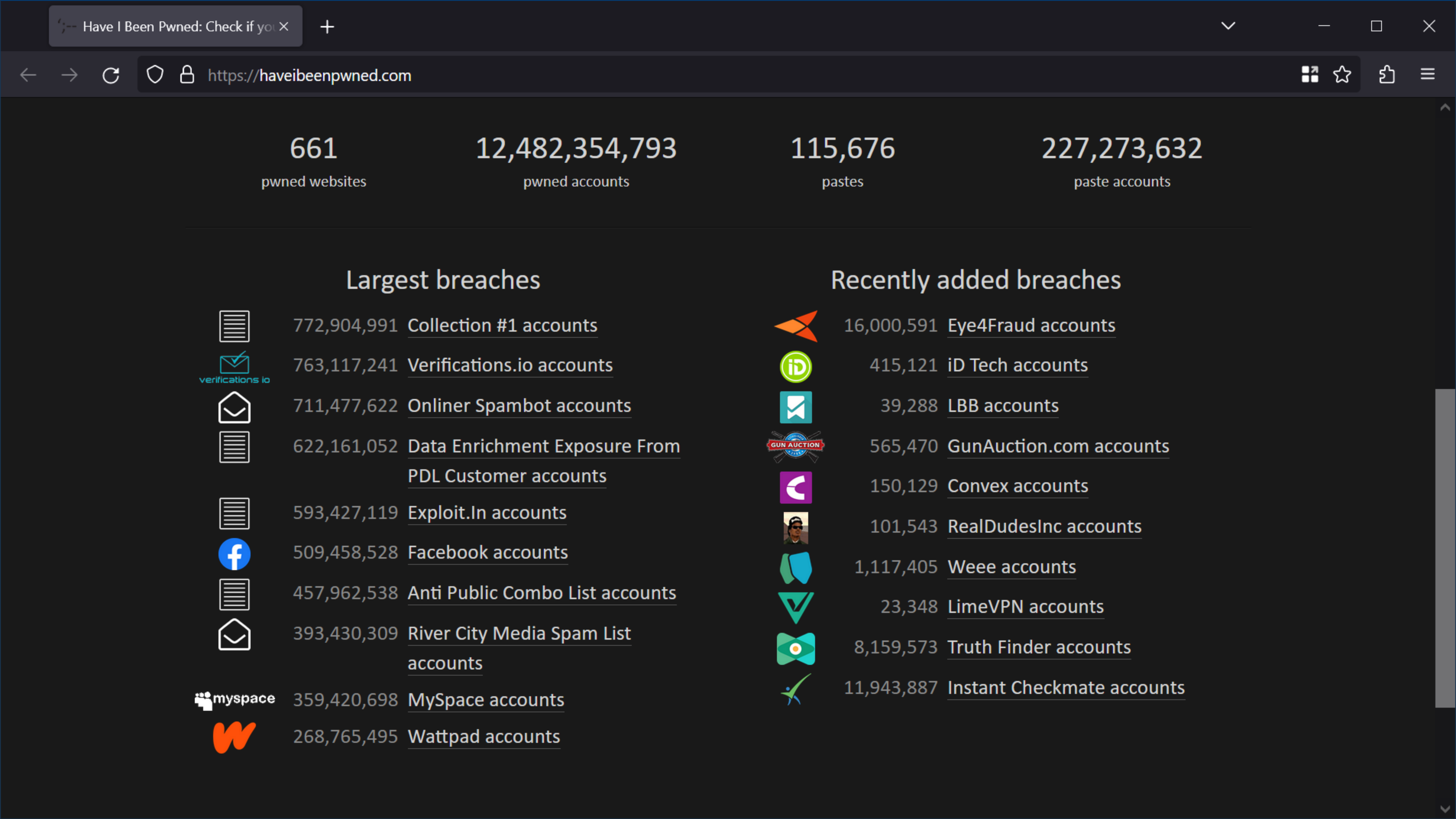
```
ce#ebc.dk
goddess5
20071002
271075711
zs3cu7za
scoopn
frygas1411
SL123456sl
12345687ee123
xuexi2010
daigoro
12345614
DICK4080
567891234
tilg80
6z08c861
:zark:
ravishsneha
150571611369
661189
passme
trolovinasveta
abdulkhaleque
007816
xLDSX
Florida2011
037037
WestC0untry
hitsugaiya
955998126
3n3rmax
4637324
bugger825
marmaris
jinjin111
170383gp
3484427
fl33321
zwqrfg
67070857
432106969
6856
704870704870
pv041886
20060814
512881535
milanimilani
472619
dbyxw888
85717221
cc841215
ariana19321
bbbnnn
ang34hehiu
wj112358
Brenda85
786525pb
shi461988
pingu
yeybozip
71477nak
stokurew
gea8mw4yz
kukumbike
260888
jordi10
lexusis
kj011a039
c84bwlrb
priyanka05
loveneverdies
u8Aqebj576
FGYfgy77
659397
327296
74748585
19720919
050769585
nicopa
2232566
bearss
n0tpublic
isitreal00
ashraf19760
48144
22471015
antyzhou115
0167005246
ec13kag
226226226226
6767537/33
mimilebrock
gueis8850
fujinshan
counter
N8mr0n
520057
adc123
bmaster
qbjh04zg
ueldaa79
EMANUELLI
yanjing
assynt
62157173
0704224950753
6903293
axaaxa
hilall
30091983
2510618981
soukuokpan
tosecondlife
p4os8m6q
015614117
acw71790
lsyljm2
2xgialdl
gaybar9
88203009
MKltyh87
quiggle
2063775206
fr3iH3it
masich
pengaiwei
coalesce
56402768
thesis
aabbcc894
marion&maxime
614850
ydz220105
584521584521
txudecp
84410545
pietro.chiara
jman1514
heryarma
39joinmam
timelapse
mwinkar
251422
willrock
YHrtfgDK
xys96exq
mercadotecnia
8s5sBEx7
0125040344
margitka
omaopa
dfTi6nh
1314520521
pixma760
pearpear
gothpunksk8er
rftaeo48
8d7R0K
5172032
aics07
34mariah
dongqinwei
samarica
cap10l4
0167387943
AE86Trueno
19700913
mcsuap
bu56mpbu
danbee
passw<>
money521
conan83
nxfjpl
rateg143
kojyihen
058336257
sarah4444
7363437
freindship
JytmvWO848
sb inbau
30907891
0515043111
1973@ati
wlxgjf
20081010
leelou44
8UfjeGbO
200358808
dellede
liang123.
captainettekt
kwiki-mart
mdovydas
tigmys2001
denial
678ad5251
woaiwuai
1591591591212
hNbDGN
cardcap
13985039393
001104
desare11
412724198
nibh1kab
asferg
hqb555
xgames7
muckerlee
choqui67
12130911
lierwei120
skytdvn
milena1995
kambala11
```


Hashes.org - Leaked Lists
Secure | https://hashes.org/public.php#

LEAKED LISTS

Complete left lists from public leaks

ID	Name	Last Update	Num of Hashes	Progress	Left Hashes	Found
6505	H4v3 1 b33n pwn3d (SHA1)	02.10.2017 - 02:03:24	320'294'464	319'837'535 (99.86%)	Get	Get
5638	P4y4sUGym (MD5)	02.10.2017 - 02:04:19	241'266	221'152 (91.66%)	Get	Get
4920	L1nk3d1n (SHA1)	02.10.2017 - 03:24:58	61'829'262	60'147'825 (97.28%)	Get	Get
3282	4mzr3v13w7r4d3r.c0m (MYSQL5)	02.10.2017 - 03:25:32	41'823	39'166 (93.65%)	Get	Get
3186	X5pl17 (SHA1)	02.10.2017 - 03:32:38	2'227'254	2'162'101 (97.07%)	Get	Get
2499	Hashkiller 32-hex left total	02.10.2017 - 11:48:14	9'976'651	1'723'709 (17.28%)	Get	Get
2498	Hashkiller 40-hex left total	02.10.2017 - 13:22:34	1'739'204	350'788 (20.17%)	Get	Get
1619	4m4t3urc0mmuni7y.c0m	02.10.2017 - 13:33:26	197'302	57'407 (29.1%)	Get	Get
1535	b73r.c0m (MD5)	02.10.2017 - 13:34:43	63'070	32'543 (51.6%)	Get	Get
1427	4v17r0n.fr	02.10.2017 - 13:34:43	2'405	2'334 (97.05%)	Get	Get
1366	v0d4f0n3 (MD5($pass."s+(_a*)")	02.10.2017 - 13:34:44	322	307 (95.34%)	Get	Get



661	12,482,354,793	115,676	227,273,632
pwned websites	pwned accounts	pastes	paste accounts

Largest breaches

772,904,991	Collection #1 accounts
763,117,241	Verifications.io accounts
711,477,622	Onliner Spambot accounts
622,161,052	Data Enrichment Exposure From PDL Customer accounts
593,427,119	Exploit.In accounts
509,458,528	Facebook accounts
457,962,538	Anti Public Combo List accounts
393,430,309	River City Media Spam List accounts
359,420,698	MySpace accounts
268,765,495	Wattpad accounts

Recently added breaches

16,000,591	Eye4Fraud accounts
415,121	iD Tech accounts
39,288	LBB accounts
565,470	GunAuction.com accounts
150,129	Convex accounts
101,543	RealDudesInc accounts
1,117,405	Weee accounts
23,348	LimeVPN accounts
8,159,573	Truth Finder accounts
11,943,887	Instant Checkmate accounts

Password Hashing Functions

Hash functions are very fast to evaluate ➔ facilitate fast password cracking

Solution: slow down the guessing process (password *"stretching"*)

- Benefit: cracking becomes very inefficient (e.g., 10-100ms per check)
- Drawback: increased cost for the server if it must authenticate many users

Make heavy use of available resources

- Fast enough computation to validate honest users, but render password guessing infeasible
- Adaptable: flexible cost (time/memory complexity) parameters

Bcrypt [Provos and Mazières, 1999]

- Cost-parameterized, modified version of the Blowfish encryption algorithm
- Tunable cost parameter (exponential number of loop iterations)

Alternatives: Scrypt (memory-hard), PBKDF2 (PKCS standard)

A Future-Adaptable Password Scheme – USENIX ATC 1999

Online Guessing

Similar strategy to offline guessing, but rate-limited

- Connect, try a few passwords, get disconnected, repeat…

Prerequisite: know a valid user name

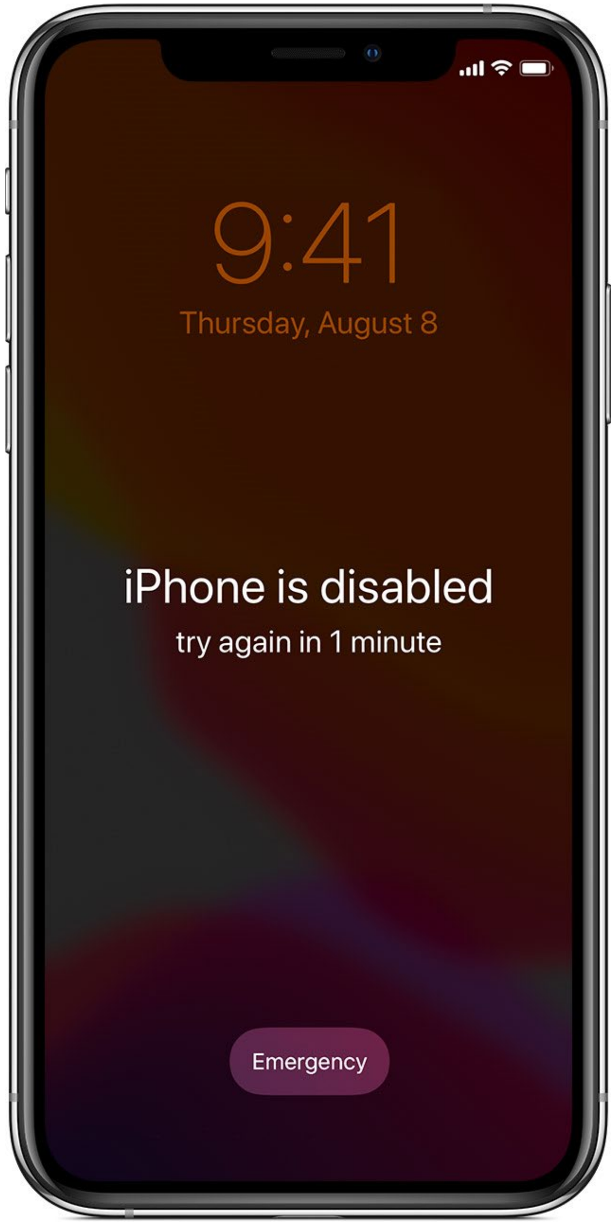
- **Credential stuffing**: try username + password combinations from previous breaches

Many failed attempts can lead to a system reaction

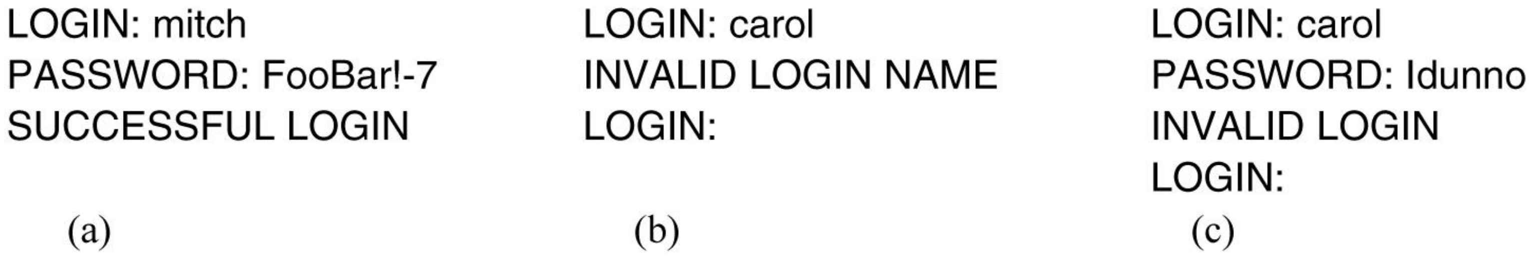
- Introduce delay before accepting future attempts (exponential backoff)
- Shut off completely (e.g., ATM capturing/disabling the card after 3 tries)
- Ask user to solve a CAPTCHA

Very common against publicly accessible SSH, VPN, RDP, and other servers

- Main reason people move `sshd` to a non-default port
- Fail2Ban: block IP after many failed attempts ➔ attackers may now be able to lock you out
- Better: disable password authentication altogether and use a key pair ➔ cumbersome if having to log in from several devices or others' computers



iPhone is disabled
try again in 1 minute



```
LOGIN: mitch
PASSWORD: FooBar!-7
SUCCESSFUL LOGIN
```

(a)

```
LOGIN: carol
INVALID LOGIN NAME
LOGIN:
```

(b)

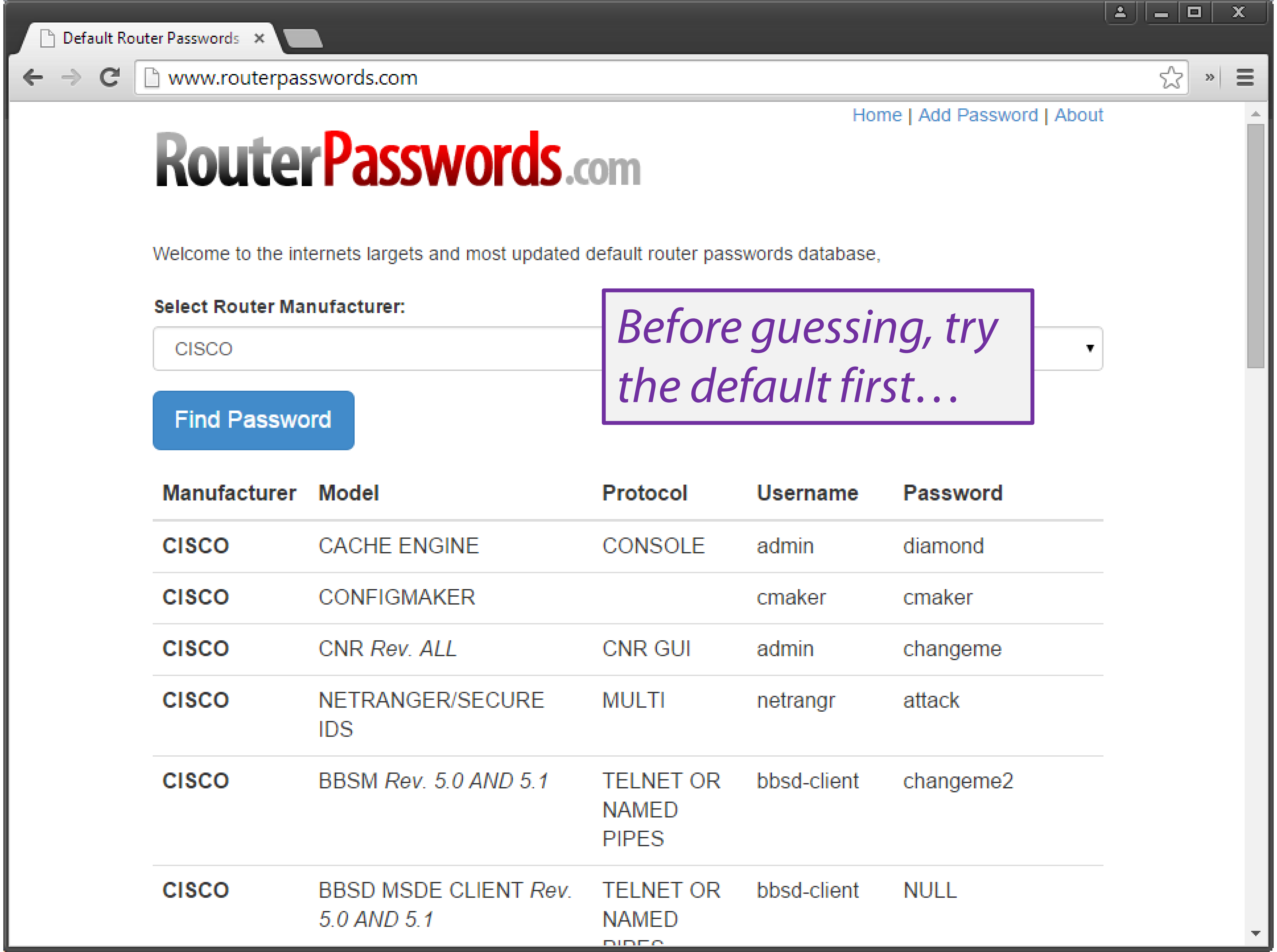
```
LOGIN: carol
PASSWORD: Idunno
INVALID LOGIN
LOGIN:
```

(c)

(a) Successful login

(b) Login rejected after name is entered

(c) Login rejected after name and password are typed ➔ less information makes guessing harder



Welcome to the internets largets and most updated default router passwords database,

Select Router Manufacturer:

CISCO

Find Password

Manufacturer	Model	Protocol	Username	Password
CISCO	CACHE ENGINE	CONSOLE	admin	diamond
CISCO	CONFIGMAKER		cmaker	cmaker
CISCO	CNR *Rev. ALL*	CNR GUI	admin	changeme
CISCO	NETRANGER/SECURE IDS	MULTI	netrangr	attack
CISCO	BBSM *Rev. 5.0 AND 5.1*	TELNET OR NAMED PIPES	bbsd-client	changeme2
CISCO	BBSD MSDE CLIENT *Rev. 5.0 AND 5.1*	TELNET OR NAMED	bbsd-client	NULL

Before guessing, try the default first…

Eavesdropping and Replay

Physical world

- Watch user type password (shoulder surfing)
- Cameras (e.g., ATM skimmers)
- Lift fingerprints (e.g., Apple Touch ID)
- Post-it notes, notebooks, …

Network makes things easier

- Sniffing (LAN, WiFi, …)
- Man-in-the-Middle attacks

Defenses

- Encryption
- One-time password schemes

Kerberos Network Authentication Protocol

- Most widely used (non-web) single sign-on system
  - Originally developed at MIT, now used in Unix, Windows, …
- Long-lived vs. session keys
  - Use long-lived key for authentication and negotiating session keys
  - Use “fresh,” ephemeral session keys for encrypted communication, MACs, …
  - Prevent replay, cryptanalysis, old compromised keys
- Authenticate users to services: using their password as the initial key, without having to retype it for every interaction
  - A Key Distribution Center (KDC) acts as a trusted third party for key distribution
  - Online authentication: variant of Needham-Schroeder protocol
  - Assumes a non-trusted network: prevents eavesdropping
  - Assumes that the Kerberos server and user workstations are secure…
- Use cases: workstation login, remote share access, printers, …

https://web.mit.edu/kerberos/

Password Capture

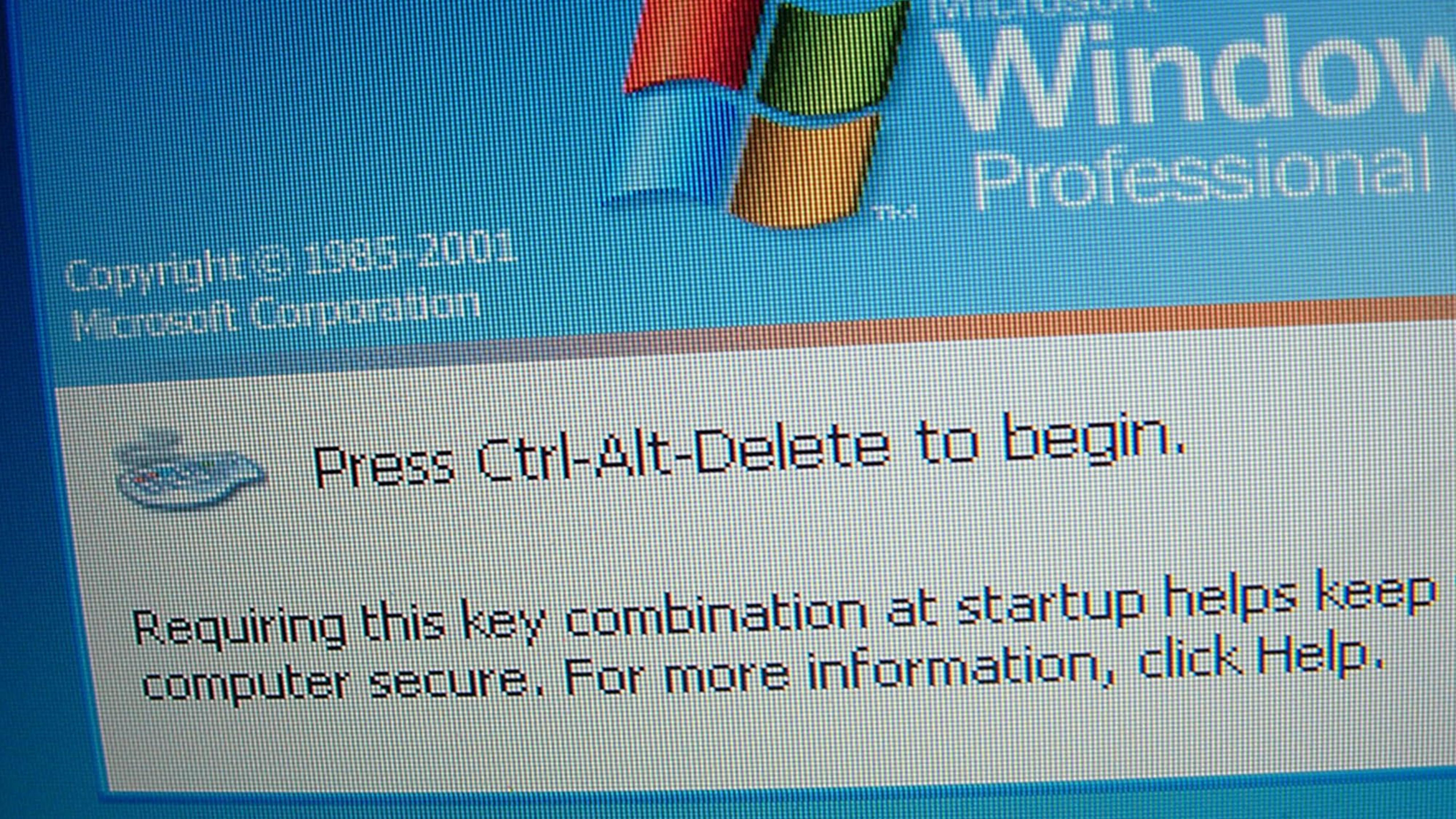
Hardware bugs/keyloggers

Software keyloggers/malware

Cameras

Phishing

Social engineering



Copyright © 1985-2001
Microsoft Corporation

Press Ctrl-Alt-Delete to begin.

Requiring this key combination at startup helps keep
computer secure. For more information, click Help.

Login:
(a)
Login:
(b)

(a) Correct login screen

(b) Phony login screen

Something You Have: Authentication Tokens

One-time passcode tokens

- Time-based or counter-based

Various other authentication tokens

- Store certificates, encryption keys, challenge–response, …

Smartcards (contact or contactless)

- Identification, authentication, data storage, limited processing
- Magnetic stripe cards, EMV (chip-n-pin credit cards), SIM cards, RFID tags, …

USB/BLE/NFC tokens, mobile phones, watches, …

- Can be used as authentication devices

Something You Are: Biometrics

Fingerprint reader

Face recognition

- Depth sensing, infrared cameras, …
- Liveness detection (pulse, thermal) to foil simple picture attack

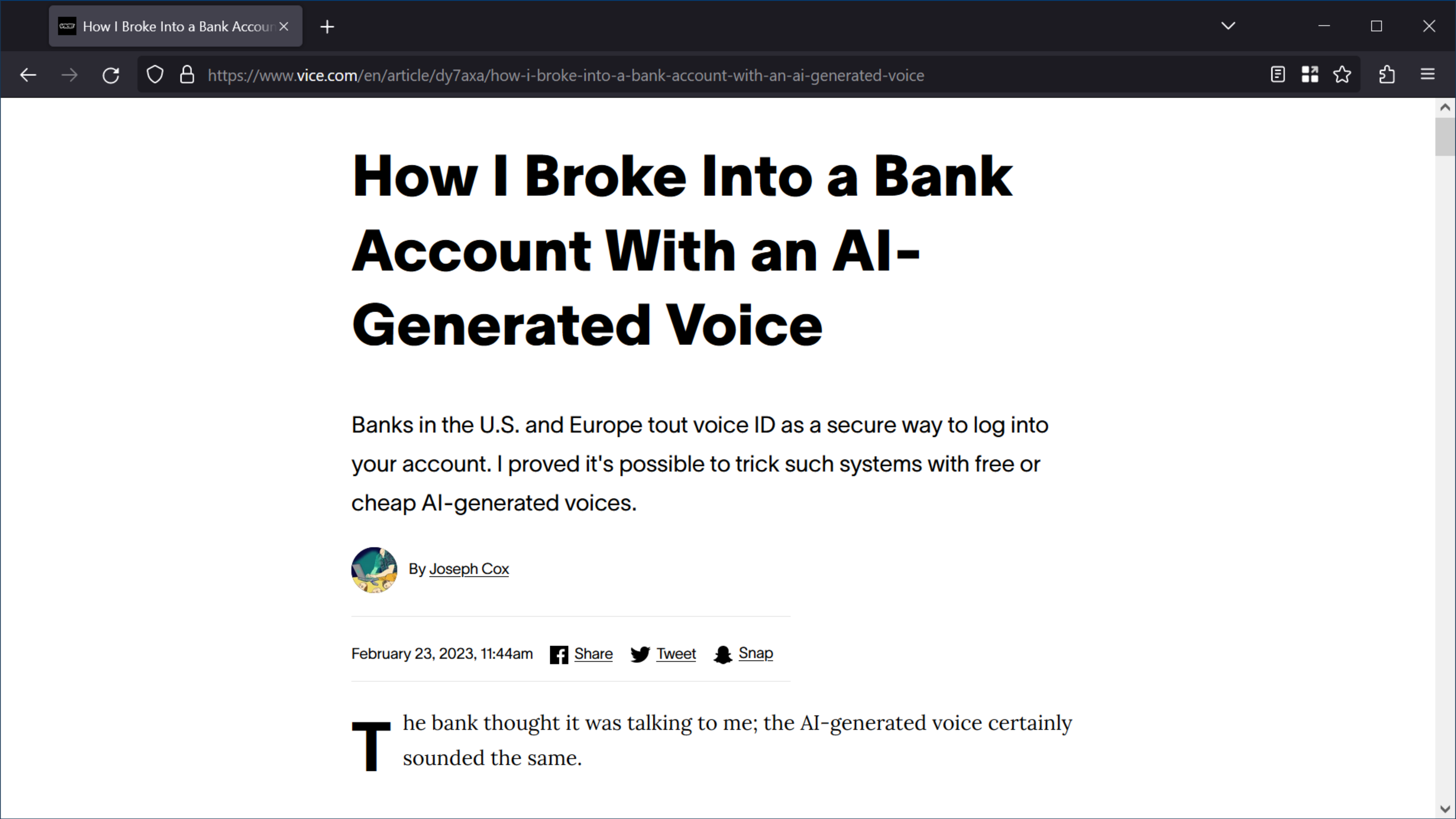
Retina/iris scanner

~~Voice recognition~~ ➔ broken

…

Related concept: continuous authentication

- Keystroke timing, usage patterns, …



How I Broke Into a Bank Accoun
https://www.vice.com/en/article/dy7axa/how-i-broke-into-a-bank-account-with-an-ai-generated-voice

How I Broke Into a Bank Account With an AI-Generated Voice

Banks in the U.S. and Europe tout voice ID as a secure way to log into your account. I proved it's possible to trick such systems with free or cheap AI-generated voices.

By Joseph Cox

February 23, 2023, 11:44am

The bank thought it was talking to me; the AI-generated voice certainly sounded the same.

Multi-factor Authentication

Must provide several separate credentials of different types

Most common: *two-factor authentication (2FA)*

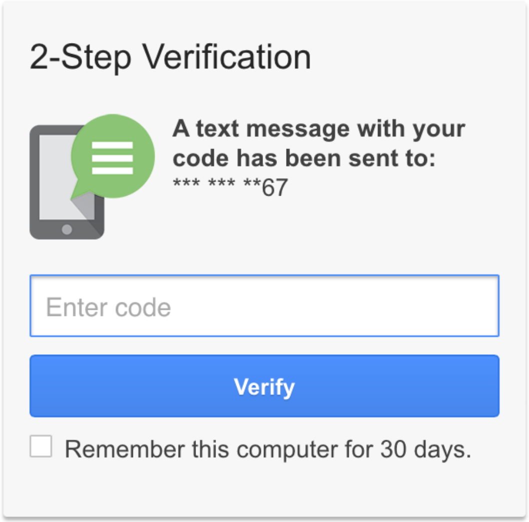
- Example: Password + hardware token/SMS message/authenticator app, …
- Example: ATM card + PIN

Motivation: a captured/cracked password is not enough to compromise a victim's account ➔ **not always true**

- *Man-in-the-Middle:* set up fake banking website, relay password to real website, let the user deal with the second factor…
- *Man-in-the-Browser:* hijack/manipulate an established session after authentication has completed (banking Trojans)
- *Dual infection:* compromise both PC and mobile device
- **More importantly: the most commonly used 2nd factor (SMS) is the least secure**

SMS Is Not a Secure 2nd Factor

(but still better than no 2nd factor)



2-Step Verification

A text message with your code has been sent to: *** *** **67

□ Remember this computer for 30 days.

Social engineering

Call victim’s mobile operator and hijack the phone number

SIM swap, message/call forwarding, …

Message interception

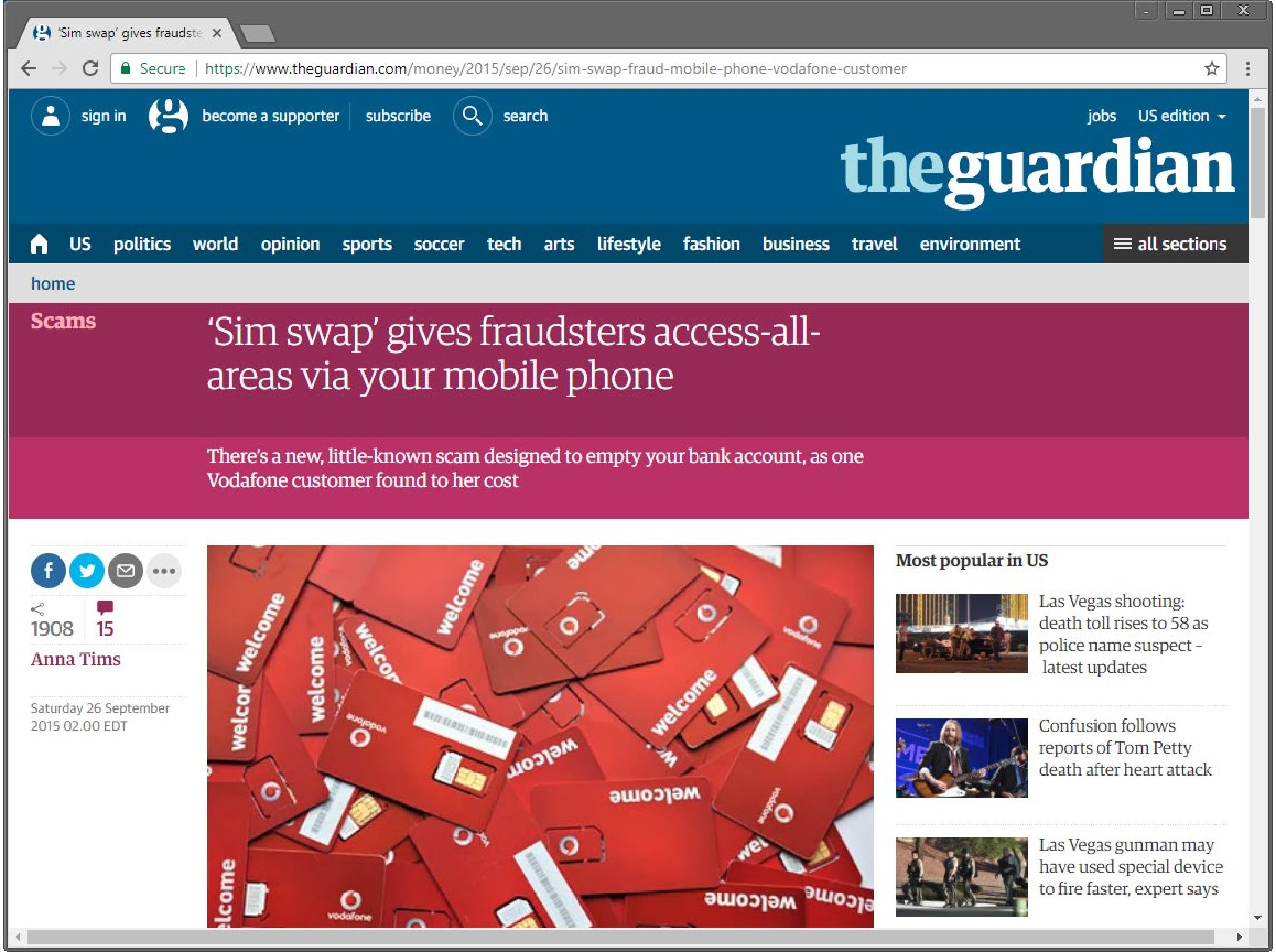
Rogue cell towers: IMSI catchers, StingRays,…

Some phones even display text messages on the lock screen (!)

SS7 attacks

The protocol used for inter-provider signaling is severely outdated and vulnerable

Allows attackers to spoof change requests to users' phone numbers and intercept calls or text messages



Scams

'Sim swap' gives fraudsters access-all-areas via your mobile phone

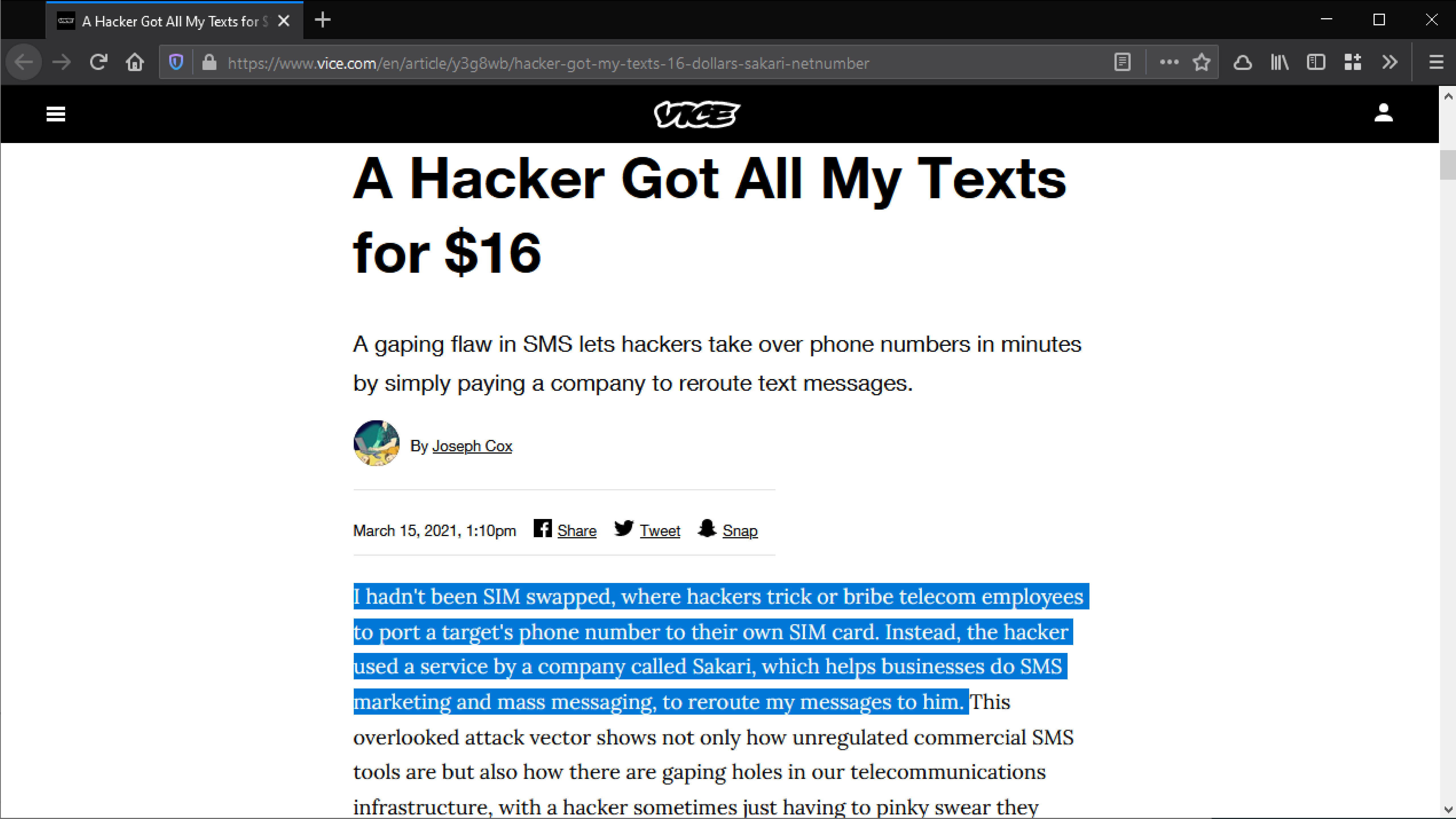
There's a new, little-known scam designed to empty your bank account, as one Vodafone customer found to her cost

Anna Tims

Saturday 26 September 2015 02.00 EDT

Most popular in US

- Las Vegas shooting: death toll rises to 58 as police name suspect - latest updates
- Confusion follows reports of Tom Petty death after heart attack
- Las Vegas gunman may have used special device to fire faster, expert says


A Hacker Got All My Texts for $
https://www.vice.com/en/article/y3g8wb/hacker-got-my-texts-16-dollars-sakari-netnumber
VICE

A Hacker Got All My Texts for $16

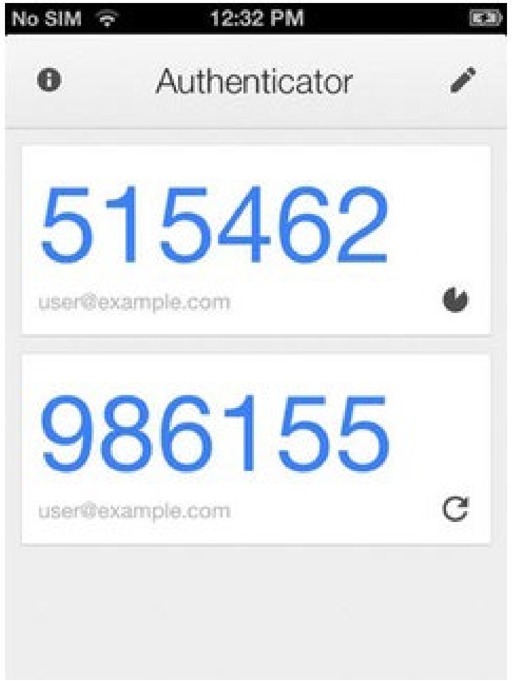
A gaping flaw in SMS lets hackers take over phone numbers in minutes by simply paying a company to reroute text messages.

By Joseph Cox

March 15, 2021, 1:10pm Share Tweet Snap

I hadn't been SIM swapped, where hackers trick or bribe telecom employees to port a target's phone number to their own SIM card. Instead, the hacker used a service by a company called Sakari, which helps businesses do SMS marketing and mass messaging, to reroute my messages to him. This overlooked attack vector shows not only how unregulated commercial SMS tools are but also how there are gaping holes in our telecommunications infrastructure, with a hacker sometimes just having to pinky swear they

Better Alternative: Authenticator App



No SIM 12:32 PM

Authenticator

515462

user@example.com

986155

user@example.com

Time-based one-time password (TOTP)

- Six/eight digit code provided after password validation
- Code computed from a shared secret key and the current time (using HMAC)
- The key is negotiated during registration

Requires “rough” client–server synchronization

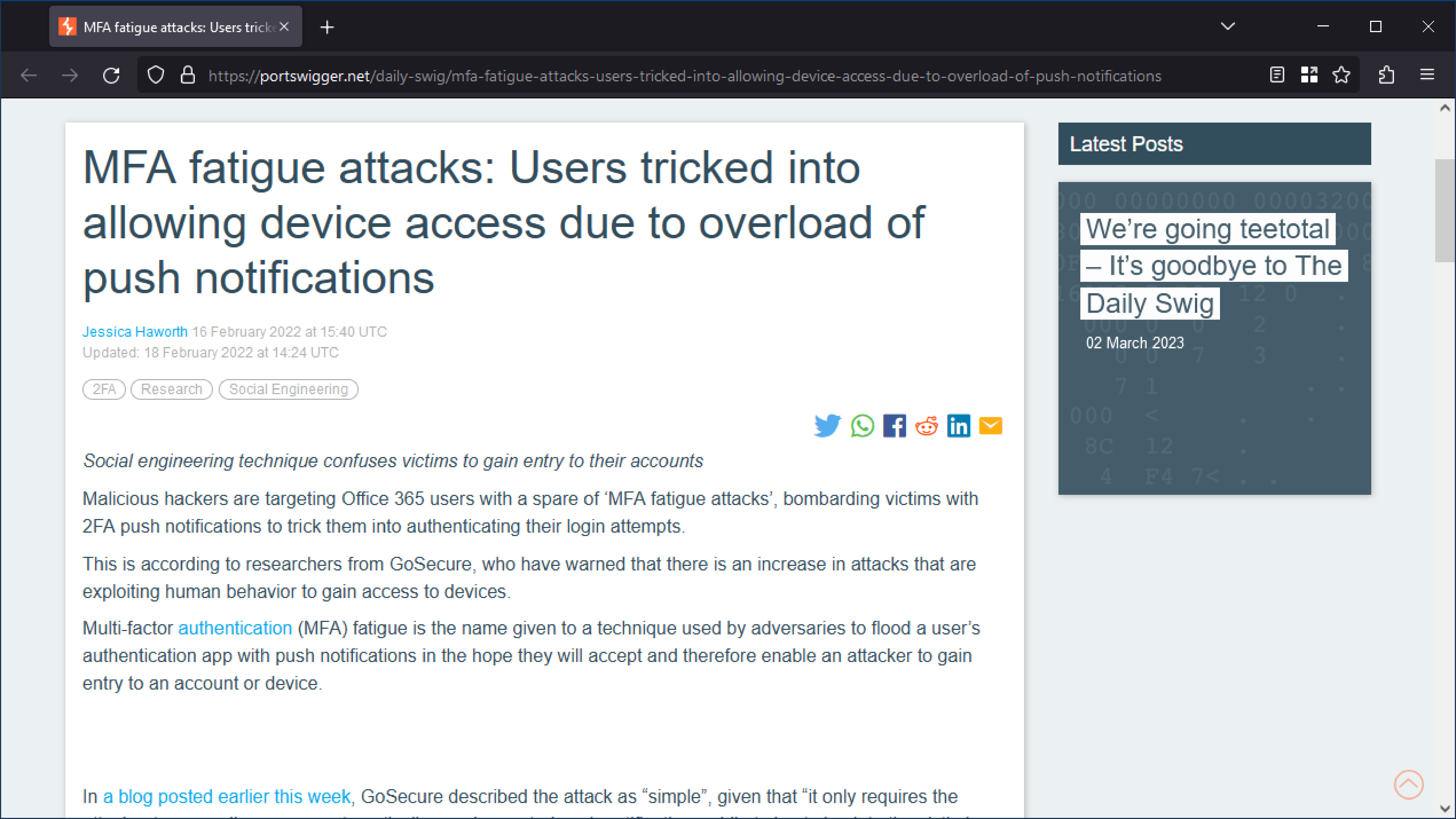
- Code constantly changes in 30-second intervals

More user-friendly alternative: push notification (e.g., Duo Push)

- MFA “fatigue” attacks: flood a user’s authentication app with push notifications

Phishing is still possible!

- The attacker just needs to proxy the captured credentials in real time (rather than collecting them for later use)


MFA fatigue attacks: Users tricke
https://portswigger.net/daily-swig/mfa-fatigue-attacks-users-tricked-into-allowing-device-access-due-to-overload-of-push-notifications

MFA fatigue attacks: Users tricked into allowing device access due to overload of push notifications

Jessica Haworth 16 February 2022 at 15:40 UTC
Updated: 18 February 2022 at 14:24 UTC

2FA Research Social Engineering

Social engineering technique confuses victims to gain entry to their accounts

Malicious hackers are targeting Office 365 users with a spare of 'MFA fatigue attacks', bombarding victims with 2FA push notifications to trick them into authenticating their login attempts.

This is according to researchers from GoSecure, who have warned that there is an increase in attacks that are exploiting human behavior to gain access to devices.

Multi-factor authentication (MFA) fatigue is the name given to a technique used by adversaries to flood a user's authentication app with push notifications in the hope they will accept and therefore enable an attacker to gain entry to an account or device.

In a blog posted earlier this week, GoSecure described the attack as "simple", given that "it only requires the

Latest Posts

We're going teetotal – It's goodbye to The Daily Swig

02 March 2023

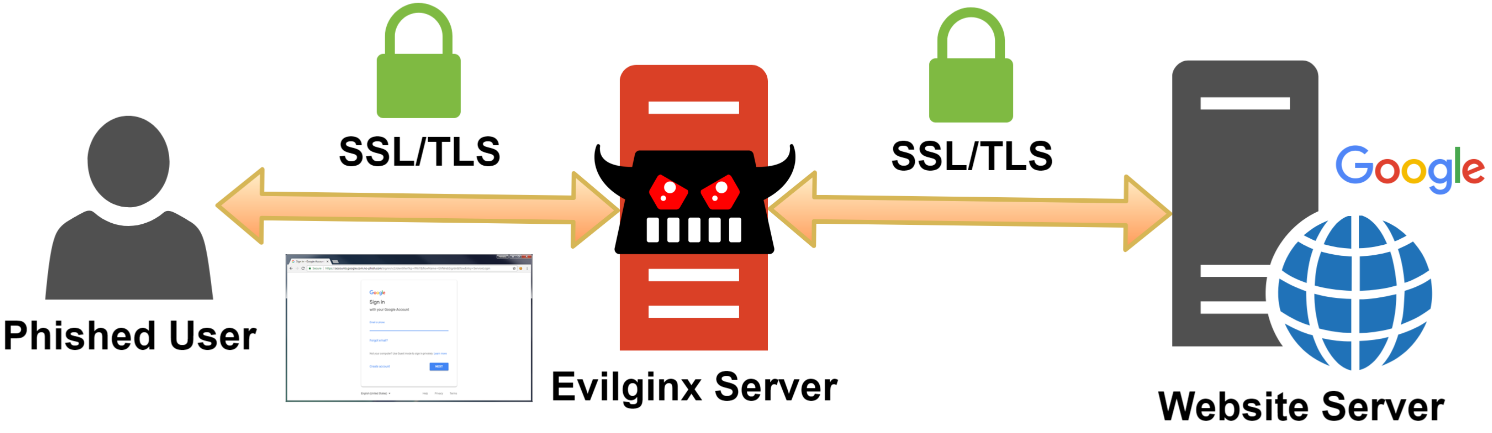
Evilginx2 https://github.com/kgretzky/evilginx2

Man-in-the-middle attack framework for phishing login credentials along with session cookies

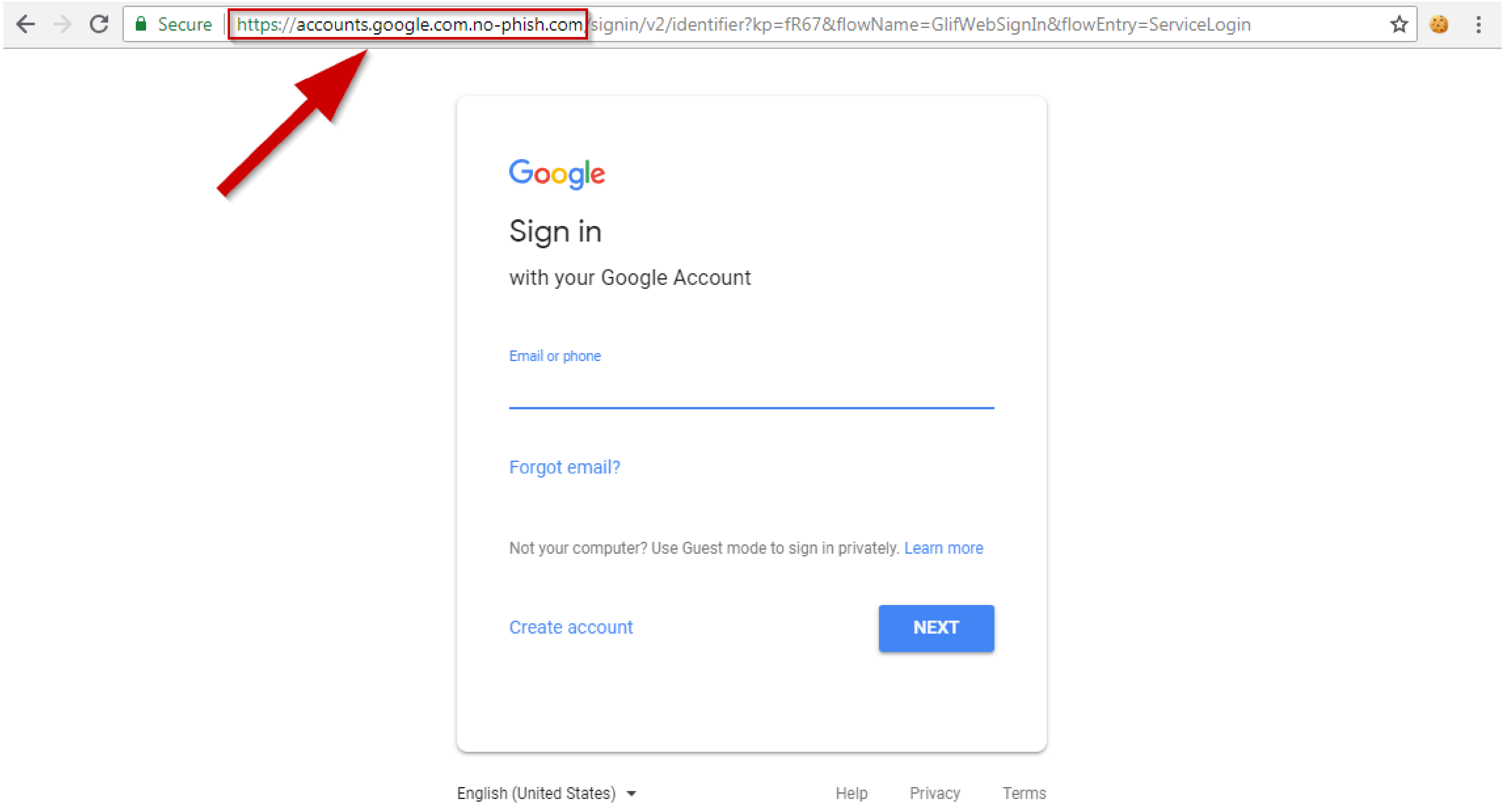
- Bypasses 2-factor authentication

No need for HTML templates: just a web proxy

- Victim's traffic is forwarded to the real website
- TLS termination at the proxy (e.g., using a LetsEncrypt certificate)



© Kuba Gretzky - https://breakdev.org/evilginx-2-next-generation-of-phishing-2fa-tokens/



Secure | https://accounts.google.com.no-phish.com/signin/v2/identifier?kp=fR67&flowName=GlifWebSignIn&flowEntry=ServiceLogin

Google

Sign in

with your Google Account

Email or phone

Forgot email?

Not your computer? Use Guest mode to sign in privately. Learn more

Create account

NEXT

English (United States) Help Privacy Terms

© Kuba Gretzky - https://breakdev.org/evilginx-2-next-generation-of-phishing-2fa-tokens/

Even Better Alternative: U2F Tokens (AKA Security Keys)

Universal Second Factor (U2F)

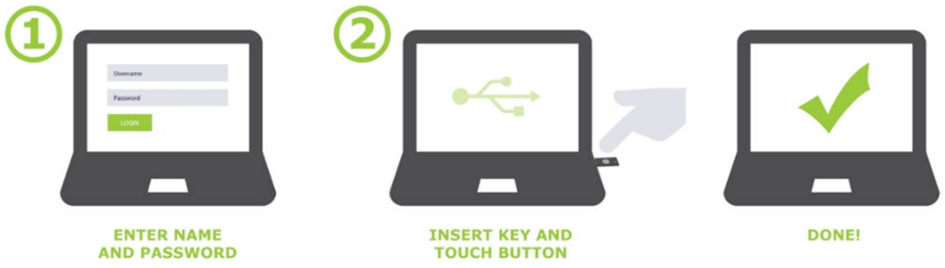
- FIDO (Fast IDentity Online) alliance: Google, Yubico, …
- Supported by all popular browsers and many online services

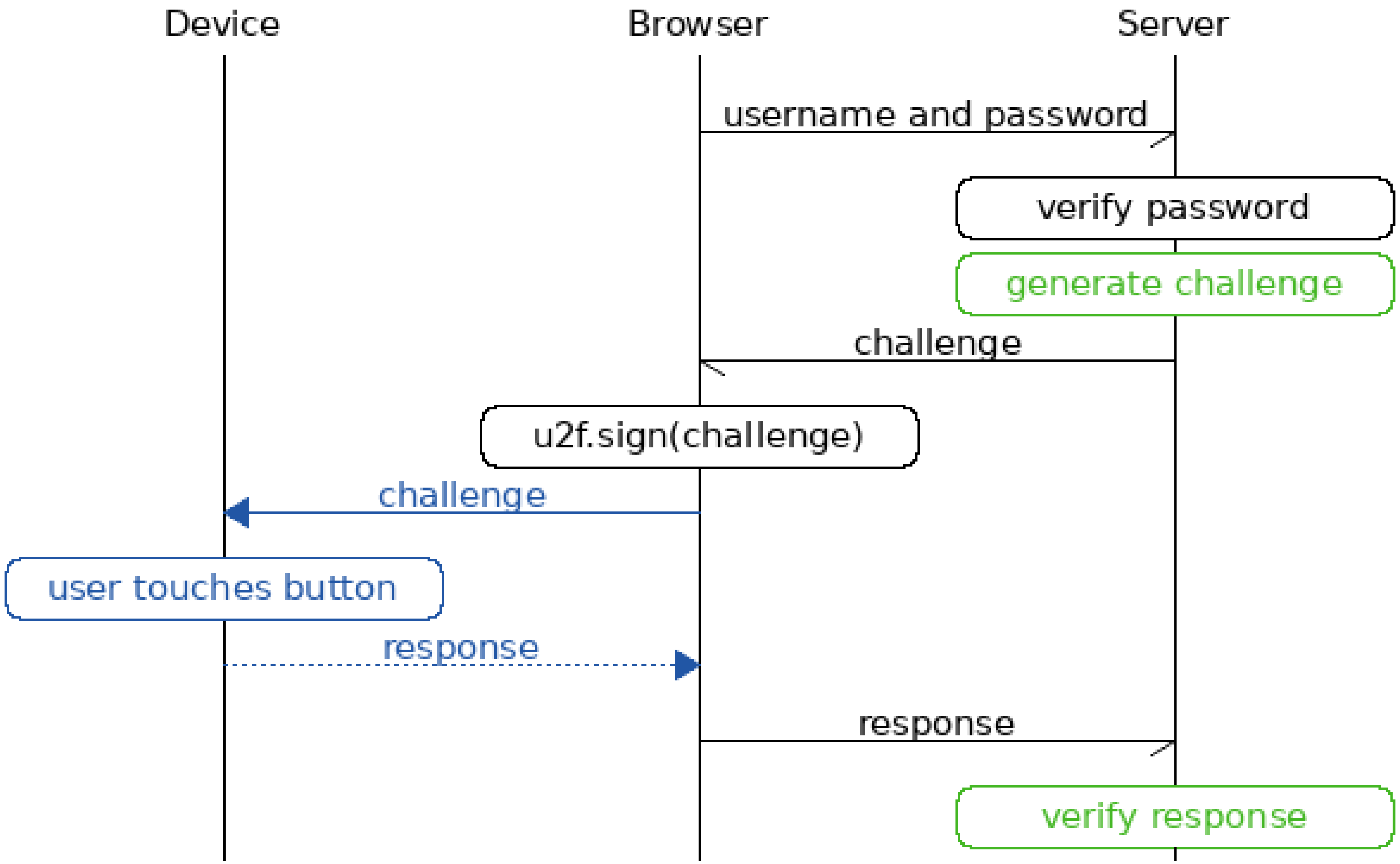
A different key pair is generated for each origin during registration

- Origin = <protocol, hostname, port>
- Private key ~~stored~~ re-generated on device
- Public key sent to server

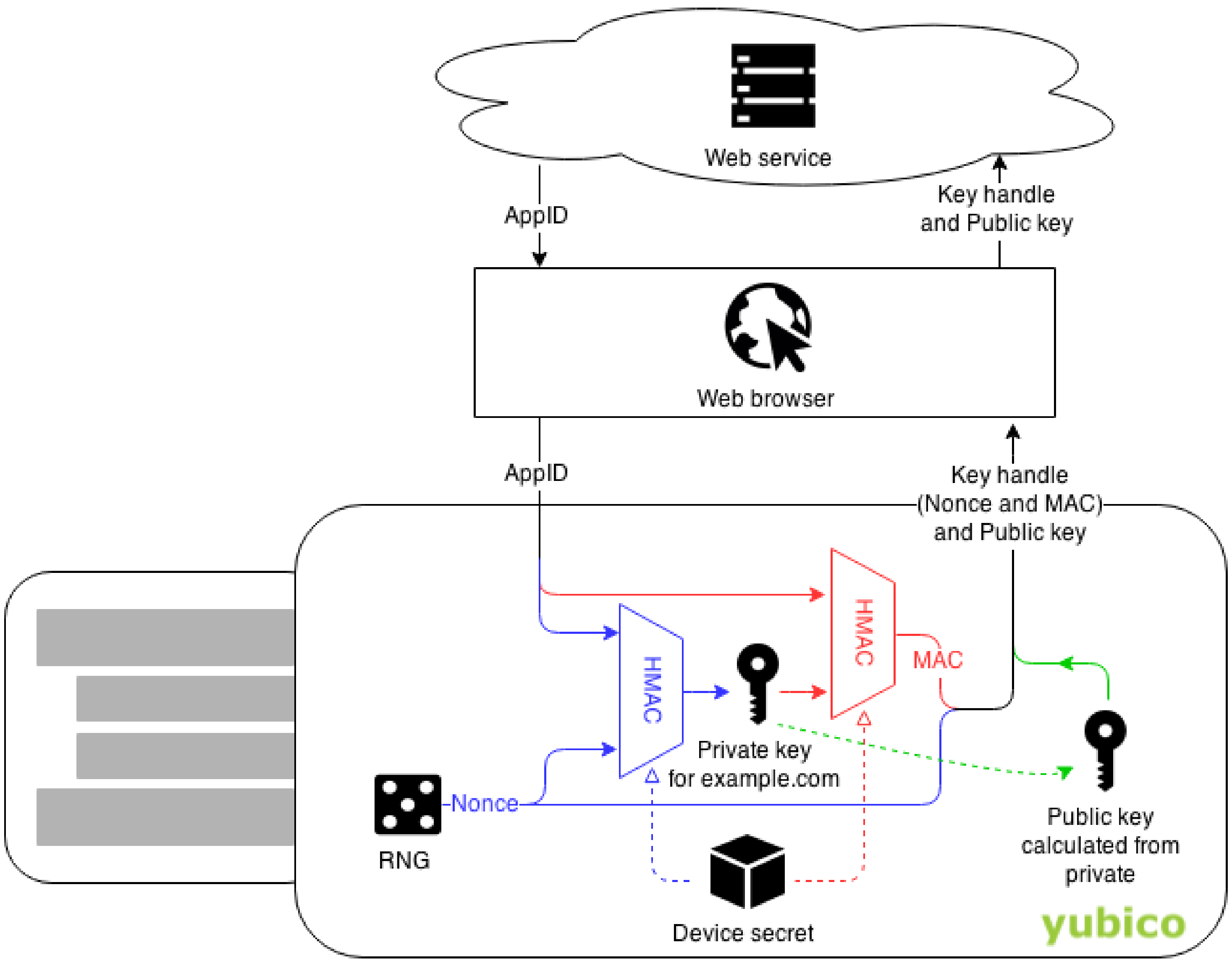
Additions to the authentication flow:

- Origin (URI): ***prevents phishing***
- TLS Channel ID (optional): *prevents MitM*





© Yubico - https://developers.yubico.com/U2F/



© Yubico - https://developers.yubico.com/U2F/Protocol_details/Key_generation.html

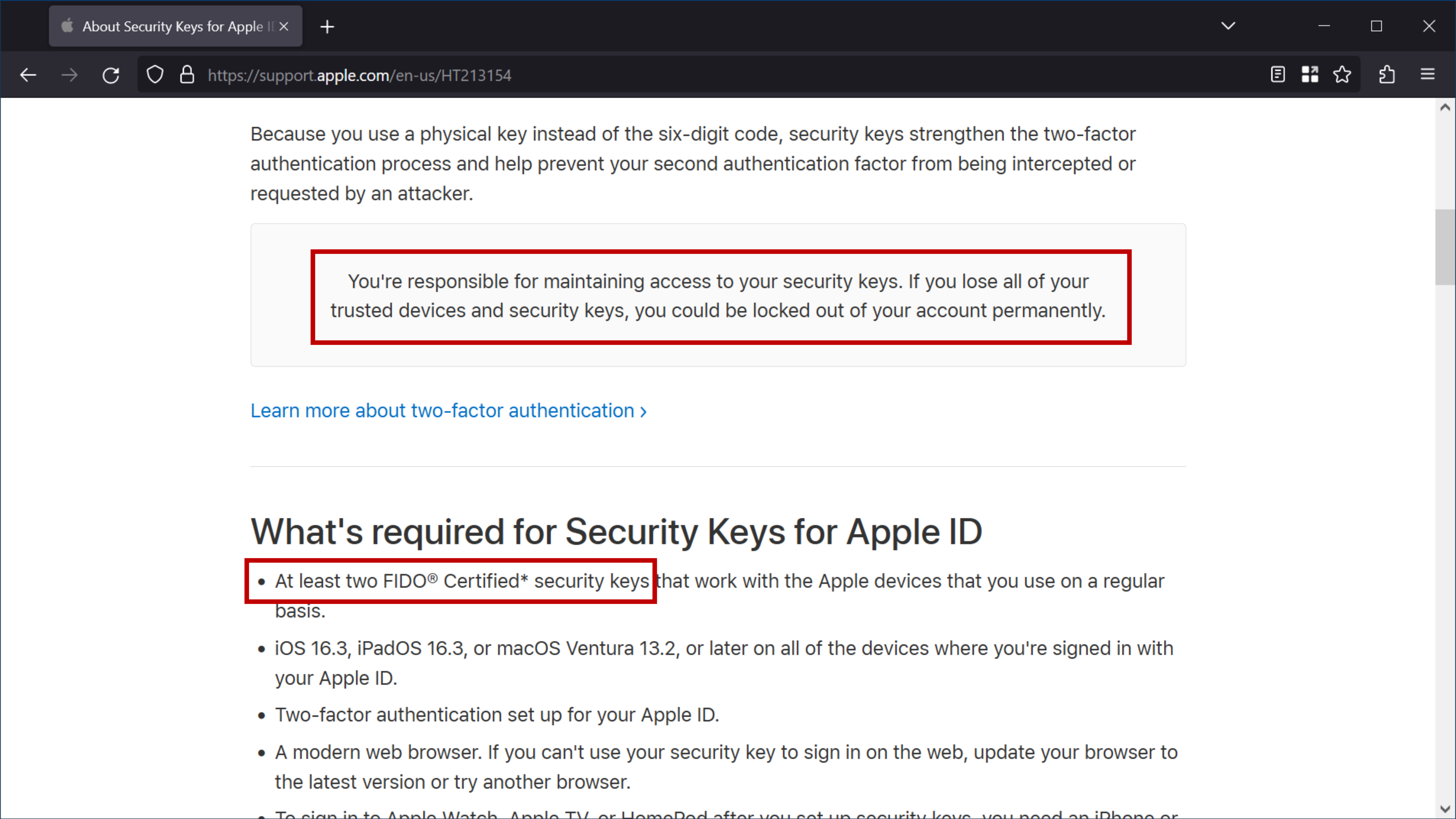
U2F tokens

Benefits

- Easy: just tap the button (no typing)
- Works out of the box (no drivers to install)
- USB, NFC, Bluetooth communication
- No shared secret between client and server
- **Origin checking ➔ effective against phishing!**

Drawbacks

- Can be lost ➔ need a fallback (backup codes, 2nd U2F token, authenticator app, …)
- Cumbersome: have to pull keychain out and plug token in (or have an always pugged-in token, in which case though it can be stolen along with the device)
- Cost ($10–$70)


About Security Keys for Apple I
https://support.apple.com/en-us/HT213154

Because you use a physical key instead of the six-digit code, security keys strengthen the two-factor authentication process and help prevent your second authentication factor from being intercepted or requested by an attacker.

You're responsible for maintaining access to your security keys. If you lose all of your trusted devices and security keys, you could be locked out of your account permanently.

Learn more about two-factor authentication ›

What's required for Security Keys for Apple ID

- At least two FIDO® Certified* security keys that work with the Apple devices that you use on a regular basis.
- iOS 16.3, iPadOS 16.3, or macOS Ventura 13.2, or later on all of the devices where you're signed in with your Apple ID.
- Two-factor authentication set up for your Apple ID.
- A modern web browser. If you can't use your security key to sign in on the web, update your browser to the latest version or try another browser.
- To sign in to Apple Watch, Apple TV, or HomePod after you set up security keys, you need an iPhone or

2FA Recap – *What threats does it prevent?*

SMS: useful against two main threats

Credential stuffing (people tend to reuse passwords across different services)

Leaked passwords (post-it, hardware keyloggers, cameras, shoulder surfing, …)

Introduces new security/privacy issues: SIM swapping, SMS forwarding, SMS spam…

Authenticator Apps/Push Auth: much better alternative than SMS

Protects against the same threats without relying on phone numbers

U2F: additional protection against phishing

Modern phishing toolkits bypass SMS/Authenticator/Push 2FA through MitM

Humans fall for typosquatting, but U2F's origin check doesn't

None of the above protect against session hijacking and Man-in-the-Browser

Game over anyway if the host is compromised after the user has successfully logged in

Password Managers

Have become indispensable

- Encourage the use of complex/non-memorable passwords
- Obviate the need for password reuse: unique passwords per site/service

Protection against phishing: auto-fill won’t work for incorrect domains

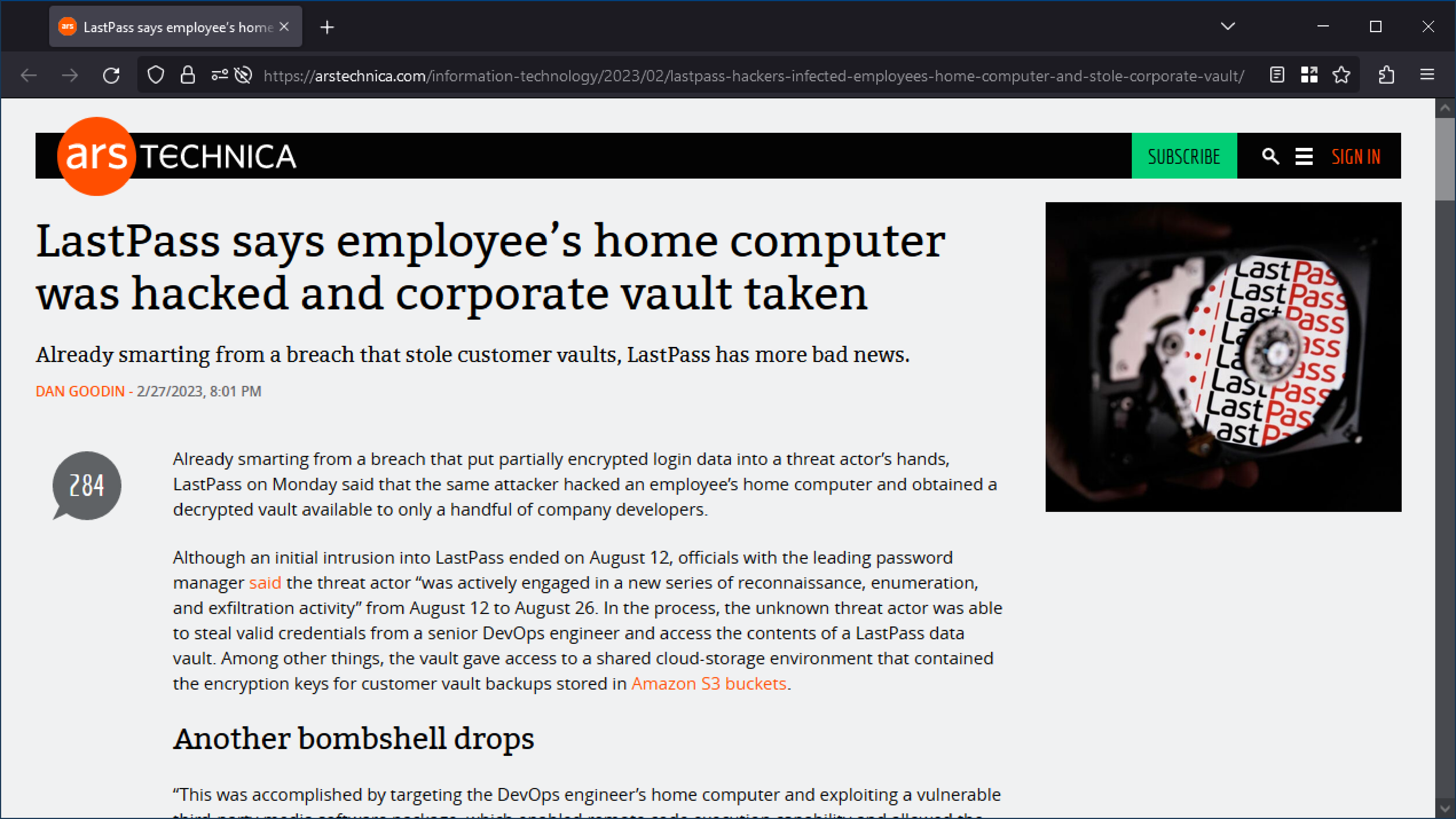
- As long as users don’t copy/paste passwords out of the password manager (!)

Various options: third-party applications, OS-level, in-browser

Password synchronization across devices

- Can the service provider access all my passwords or not?
- Preferable option: passwords should be encrypted with master password never visible to the cloud service

Single point of failure (!)


LastPass says employee's home
https://arstechnica.com/information-technology/2023/02/lastpass-hackers-infected-employees-home-computer-and-stole-corporate-vault/
ars TECHNICA
SUBSCRIBE
SIGN IN
284
LastPass LastPass LastPass

LastPass says employee's home computer was hacked and corporate vault taken

Already smarting from a breach that stole customer vaults, LastPass has more bad news.

DAN GOODIN - 2/27/2023, 8:01 PM

Already smarting from a breach that put partially encrypted login data into a threat actor's hands, LastPass on Monday said that the same attacker hacked an employee's home computer and obtained a decrypted vault available to only a handful of company developers.

Although an initial intrusion into LastPass ended on August 12, officials with the leading password manager said the threat actor "was actively engaged in a new series of reconnaissance, enumeration, and exfiltration activity" from August 12 to August 26. In the process, the unknown threat actor was able to steal valid credentials from a senior DevOps engineer and access the contents of a LastPass data vault. Among other things, the vault gave access to a shared cloud-storage environment that contained the encryption keys for customer vault backups stored in Amazon S3 buckets.

Another bombshell drops

"This was accomplished by targeting the DevOps engineer's home computer and exploiting a vulnerable third-party media software package, which enabled remote code execution capability and allowed the

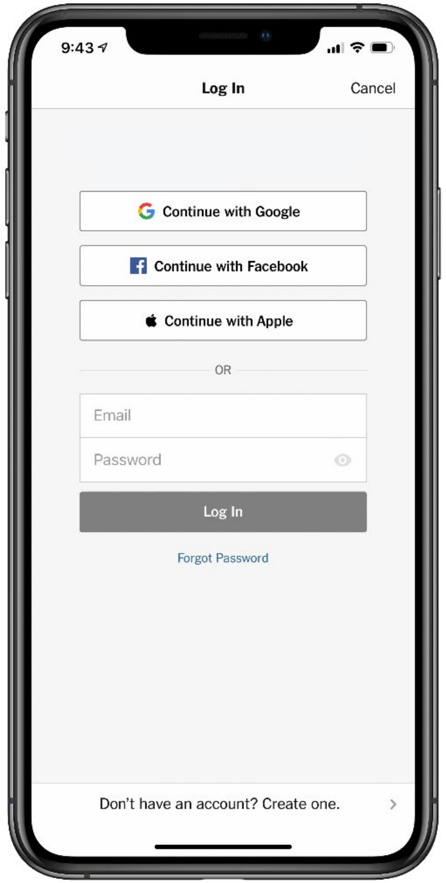
Single Sign-on/Social Login

Pros

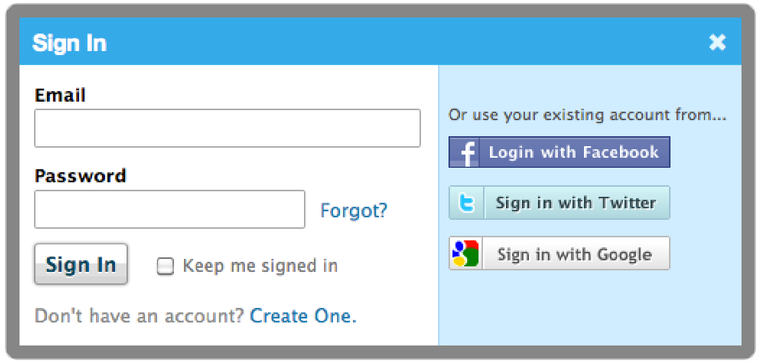
- Convenience: fewer passwords to remember
- Easier development: outsource user registration/management
- Rich experience through social features

Cons

- Same credentials for multiple sites: single point of failure
- Third-parties gain access to users' profiles
- Provider can track users



Don't have an account? Create one.



Or use your existing account from...

Don't have an account? Create One.

WebAuthn

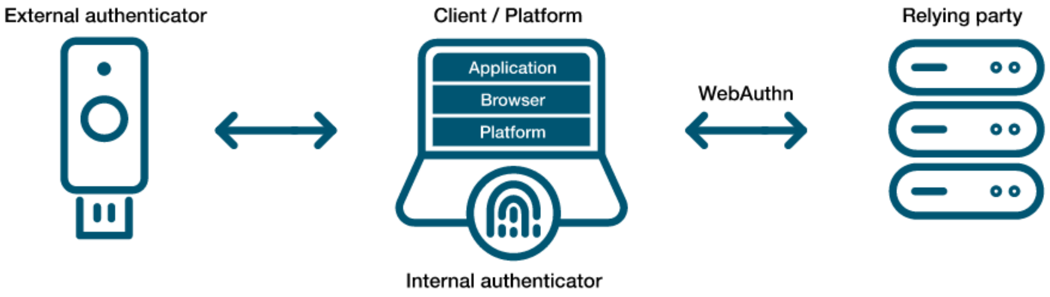
W3C Web Authentication standard (FIDO2): Successor of FIDO U2F

Use cases

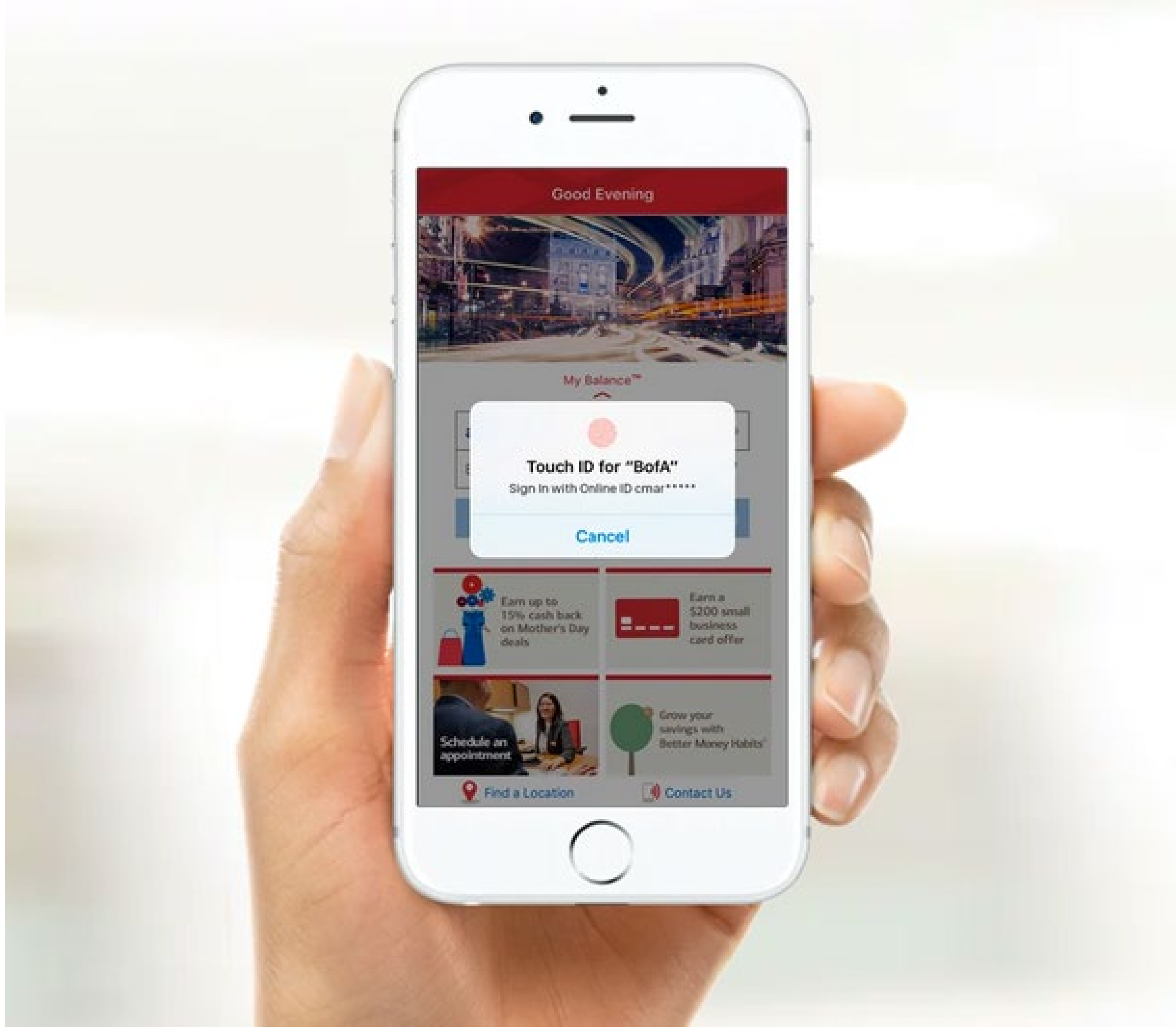
- Low friction and phishing-resistant 2FA (in conjunction with a password)
- Passwordless, biometrics-based re-authorization
- 2FA *without* a password ("passwordless" login)

Authenticators: devices that can generate private/public key pairs and gather consent (simple tap, fingerprint read, …)

- Roaming Authenticators:
  USB/BLE/NFC security keys
- Platform Authentications:
  Built-in fingerprint readers, cameras, …



https://www.yubico.com/authentication-standards/webauthn/



Touch ID for "BofA"

Sign In with Online ID cmar*****

Good Evening
My Balance™
Cancel
Earn up to 15% cash back on Mother's Day deals
Earn a $200 small business card offer
Schedule an appointment
Grow your savings with Better Money Habits®
Find a Location
Contact Us

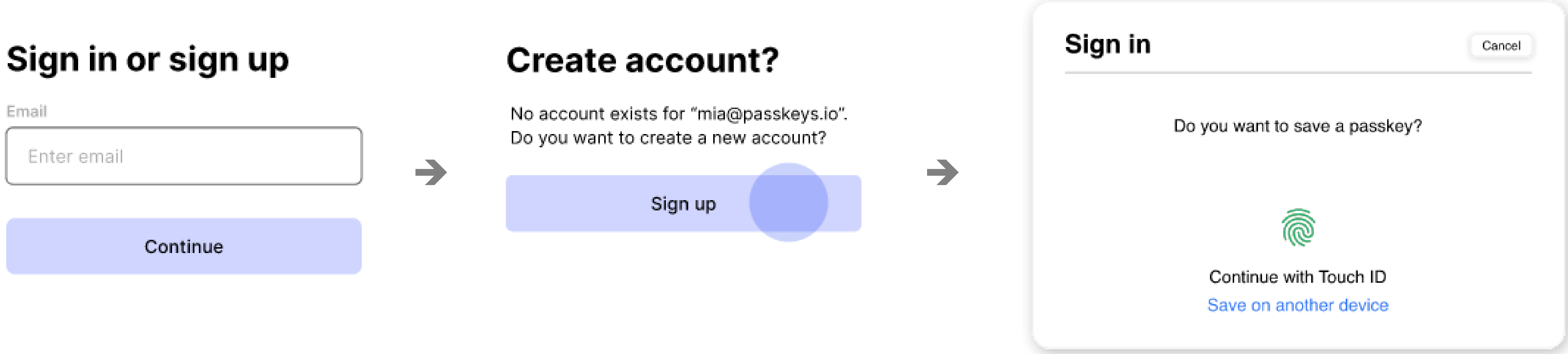
Passkeys

Completely replace passwords with cryptographic key pairs

- Server only keeps a user's public key
- Based on WebAuthn: rely on biometric identification (Touch ID, Face ID)

Key enabler: identity providers who also sell hardware devices

- The user's device becomes an authenticator → what if it gets lost?
- Users have more than one device → seamless syncing



Sign in or sign up

Create account?

No account exists for "mia@passkeys.io".
Do you want to create a new account?

Sign in

Do you want to save a passkey?

https://www.passkeys.io/

Multi-factor vs. Multi-step

Factor: something you know/have/are

Step: user-specific action

Type password, tap fingerprint reader, press security key, …

Example: U2F flow with passwords

Type password + tap security key ➔ two factors, two steps

********* +

Example: FIDO2 passwordless flow

Tap biometric security key ➔ two factors, one step

https://medium.com/webauthnworks/its-single-step-not-factor-clarifying-more-fido-terminology-d06d9c31b4f2

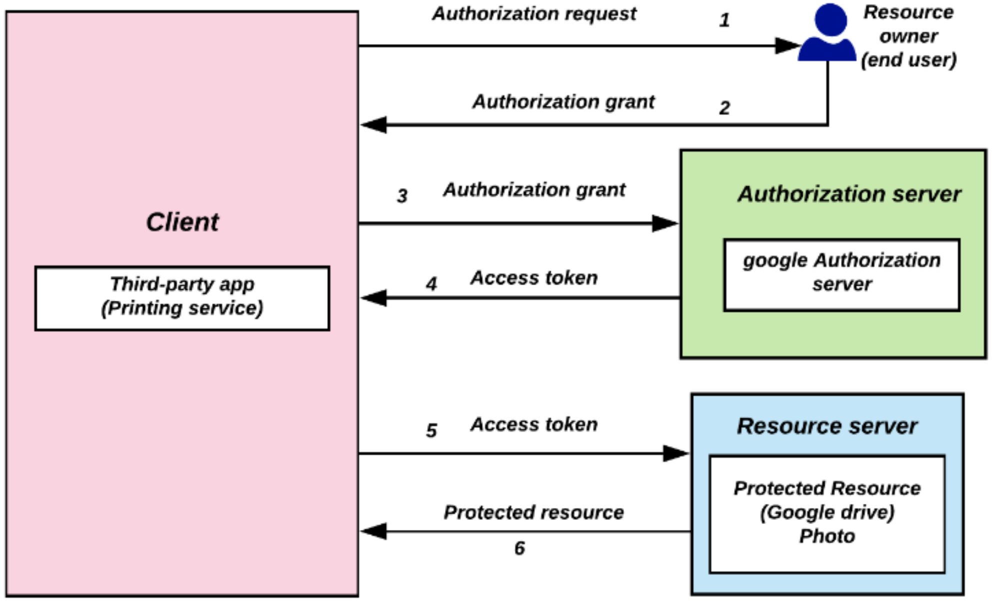
OAuth 2.0

Open standard for secure delegated access *(not authentication)*

- Allows users to grant third-party websites/apps access to their information
- Improved security: access tokens are short-lived and can be revoked at any time
- Reduced friction: users don't need to share their credentials with third parties

OAuth 2.0 Flow

- Client requests authorization from the resource owner ➔ owner grants/denies
- Client obtains an *authorization grant*
- Client exchanges the authorization grant for an *access token* from the auth server
- Client uses the access token to access protected resources from the server



https://en.wikipedia.org/wiki/OAuth

Recap: Crypto-based Authentication

Rely on a cryptographic key to prove a user's identity

User performs a requested cryptographic operation on a value (challenge) that the verifier supplies

- Usually based on knowledge of a key (secret key or private key)
- Can use symmetric (e.g., Kerberos) or public key (e.g., U2F) schemes

How can we trust a key? Why is it authentic?

- Need to establish a level of trust

Different approaches: **TOFU**, **PKI**, **Web of Trust**

Trust on First Use (aka Key Continuity)

Use case: SSH

- Performs *mutual authentication*

Server *always* authenticates the client

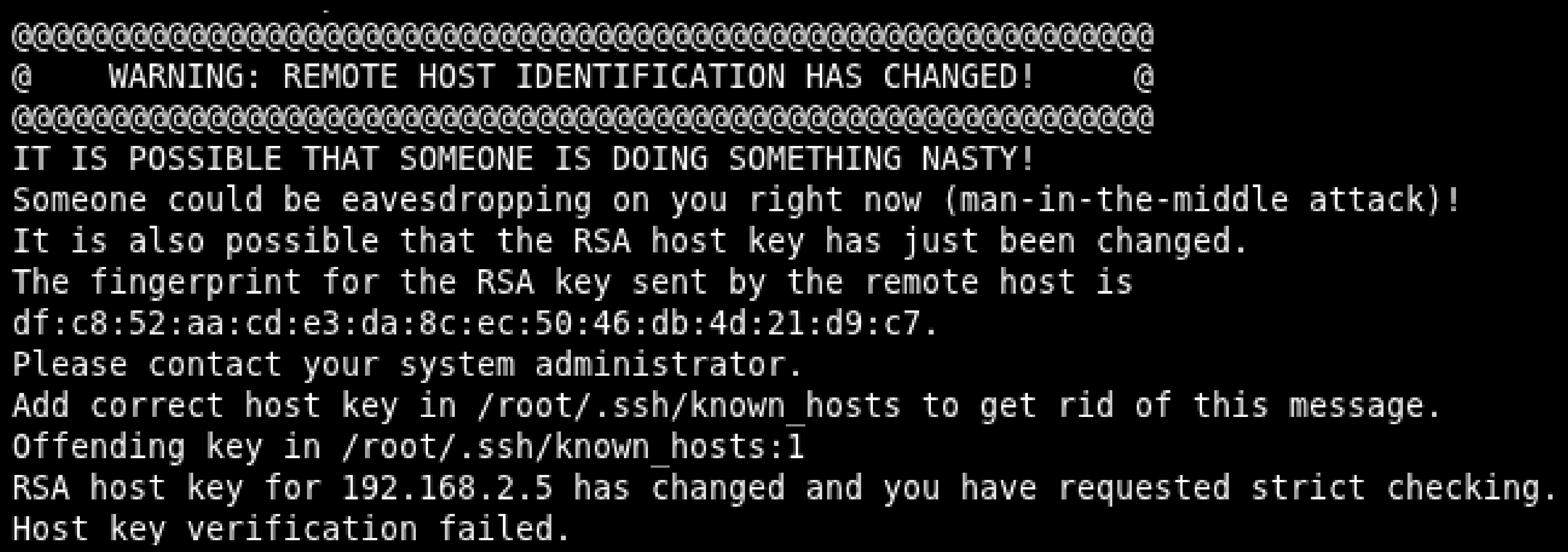
- password, key pair, …

Client *almost* always authenticates the server – *except the first time!*

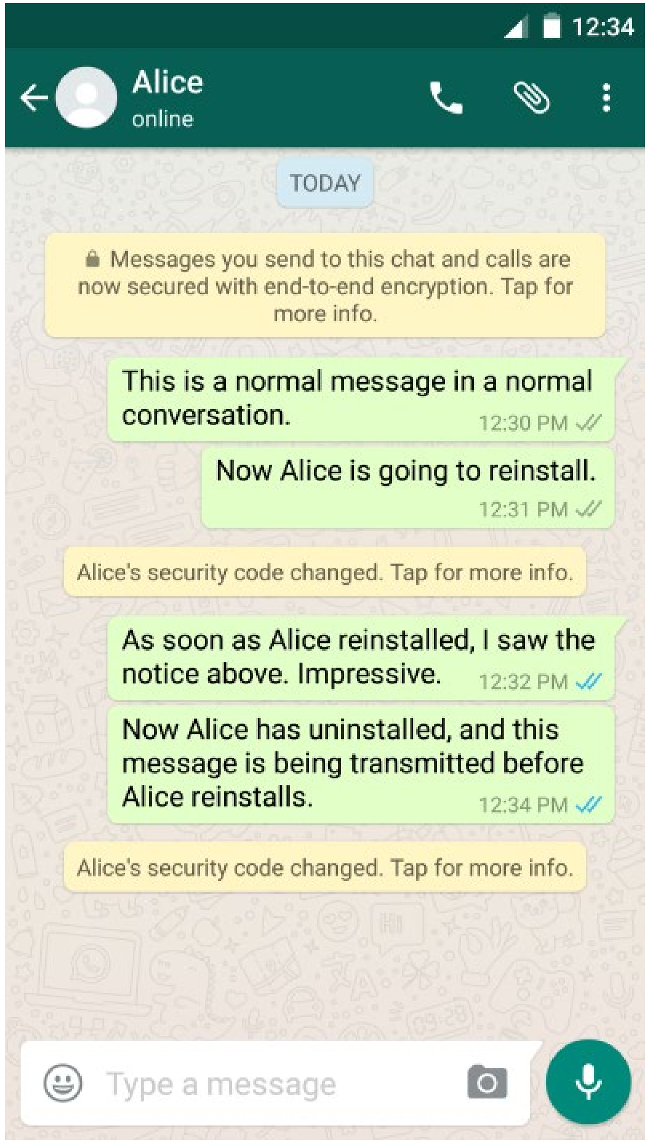
- First connection: server presents its public key
- No other option for the user but to accept it: MitM opportunity
- Subsequent connections: client remembers server’s key, and triggers an alert on key mismatch

Pragmatic solution, but shifts the burden to users

- Users must determine the validity of the presented key
- Accepting a key change without verifying the new key offers no protection against MitM (unfortunately, that’s what most users do)



```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
df:c8:52:aa:cd:e3:da:8c:ec:50:46:db:4d:21:d9:c7.
Please contact your system administrator.
Add correct host key in /root/.ssh/known_hosts to get rid of this message.
Offending key in /root/.ssh/known_hosts:1
RSA host key for 192.168.2.5 has changed and you have requested strict checking.
Host key verification failed.
```



TODAY

Messages you send to this chat and calls are now secured with end-to-end encryption. Tap for more info.

This is a normal message in a normal conversation. 12:30 PM

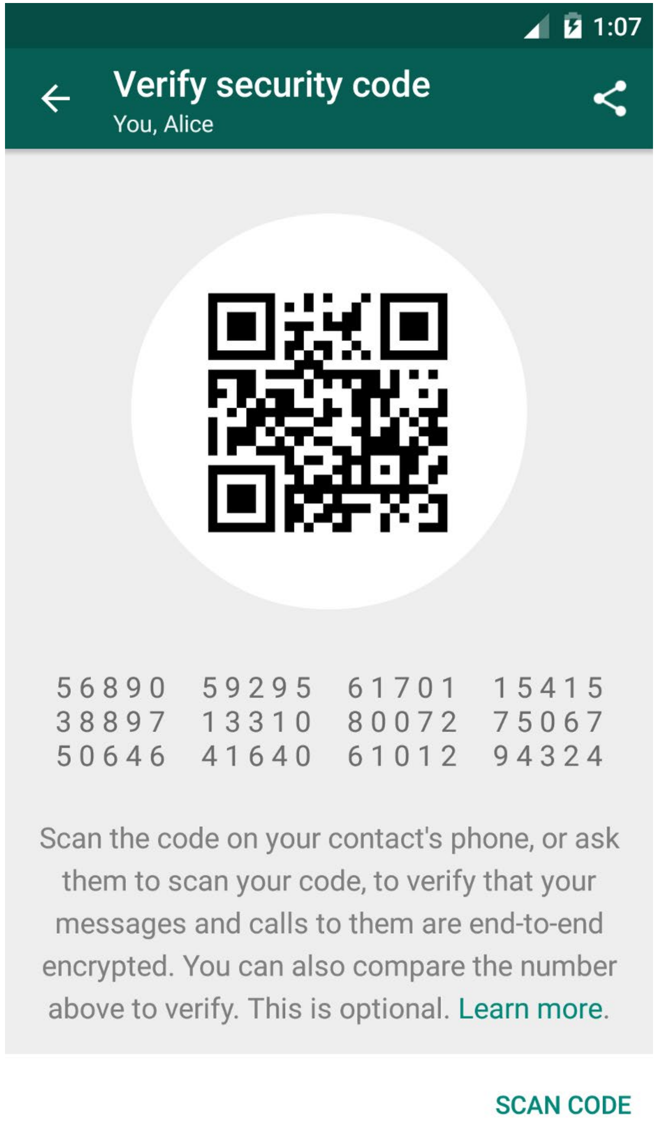
Now Alice is going to reinstall. 12:31 PM

Alice's security code changed. Tap for more info.

As soon as Alice reinstalled, I saw the notice above. Impressive. 12:32 PM

Now Alice has uninstalled, and this message is being transmitted before Alice reinstalls. 12:34 PM

Alice's security code changed. Tap for more info.



Verify security code
You, Alice

```
56890  59295  61701  15415
38897  13310  80072  75067
50646  41640  61012  94324
```

Scan the code on your contact's phone, or ask them to scan your code, to verify that your messages and calls to them are end-to-end encrypted. You can also compare the number above to verify. This is optional. Learn more.

© Tina Membe – https://medium.com/@pepelephew/a-look-at-how-private-messengers-handle-key-changes-5fd4334b809a

Certificates

How can we distribute "trusted" public keys?

- Public directory ➔ risk of forgery and tampering
- More practical solution: "certified" public keys

A certificate is a digitally signed message that contains an identity and a public key

- Makes an association between a user/entity and a private key
- Valid until a certain period
- Most common format: X.509

Why trust a certificate?

- Because it is signed by an "authority"
- Requiring a signature by a third party prevents straightforward tampering

Public Key Infrastructures (PKI)

Facilitate the authentication and distribution of public keys with the respective identities of entities

- People, organizations, devices, applications, …
- Set of roles, policies, hardware, software, and procedures to create, mange, distribute, use, store, and revoke digital certificates and manage public key encryption

An issuer signs certificates for subjects

- Trust anchor

Methods of certification

- **Certificate authorities** (hierarchical structure – root of trust)
- **Web of trust** (decentralized, peer-to-peer structure)

Certificate Authorities

Trusted third-parties responsible for certifying public keys

- Most CAs are tree-structured

A public key for any website in the world will be accepted without warning if it has been certified by a trusted CA

- Single point of failure: CAs can be compromised!

Why should we trust an authority?

- How do we know the public key of the Certificate Authority?

CA’s public key (trust anchor) must somehow be provided out of band

- Trust has to start somewhere

Certificate Chains

Trust anchors: operating systems and browsers are pre-configured with trusted root certificates

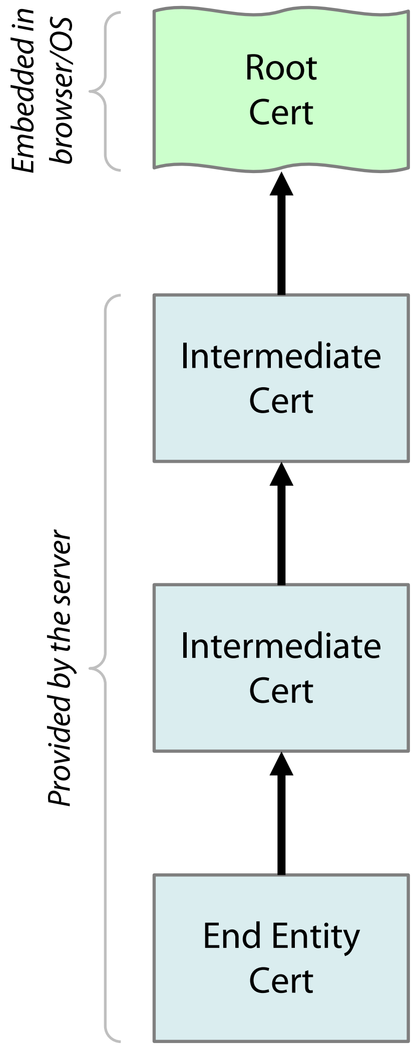
- System/public store: used by OS, browsers, …
- More can be added in the local/private cert store: vendor-specific certs, MitM certs for content inspection filters/AVs, …

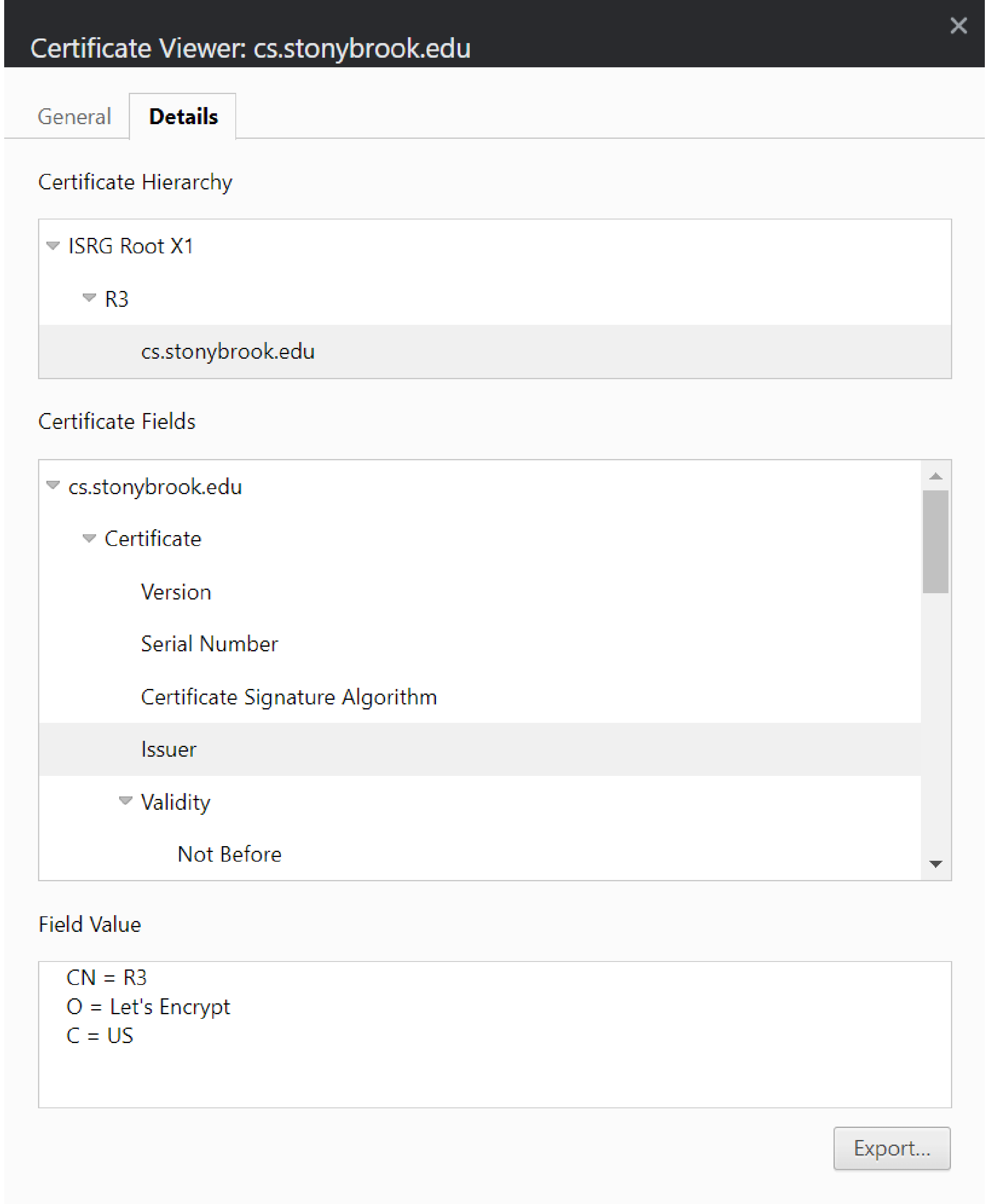
Server provides a *chain* of certificates

- A certificate from an intermediate CA is trusted if there is a valid chain of trust all the way back to a trusted root CA

Any CA can issue and sign certificates for any subject

- The system is only as secure as the weakest certificate authority…
- Certificate Authority Authorization (CAA): can be used to restrict which CAs can issue certificates for a particular domain





Certificate Viewer: cs.stonybrook.edu

Certificate Hierarchy

- ISRG Root X1
  - R3
    - cs.stonybrook.edu

Certificate Fields

- cs.stonybrook.edu
  - Certificate
    - Version
    - Serial Number
    - Certificate Signature Algorithm
    - Issuer
    - Validity
      - Not Before

Field Value

CN = R3
O = Let's Encrypt
C = US

Certificate Revocation

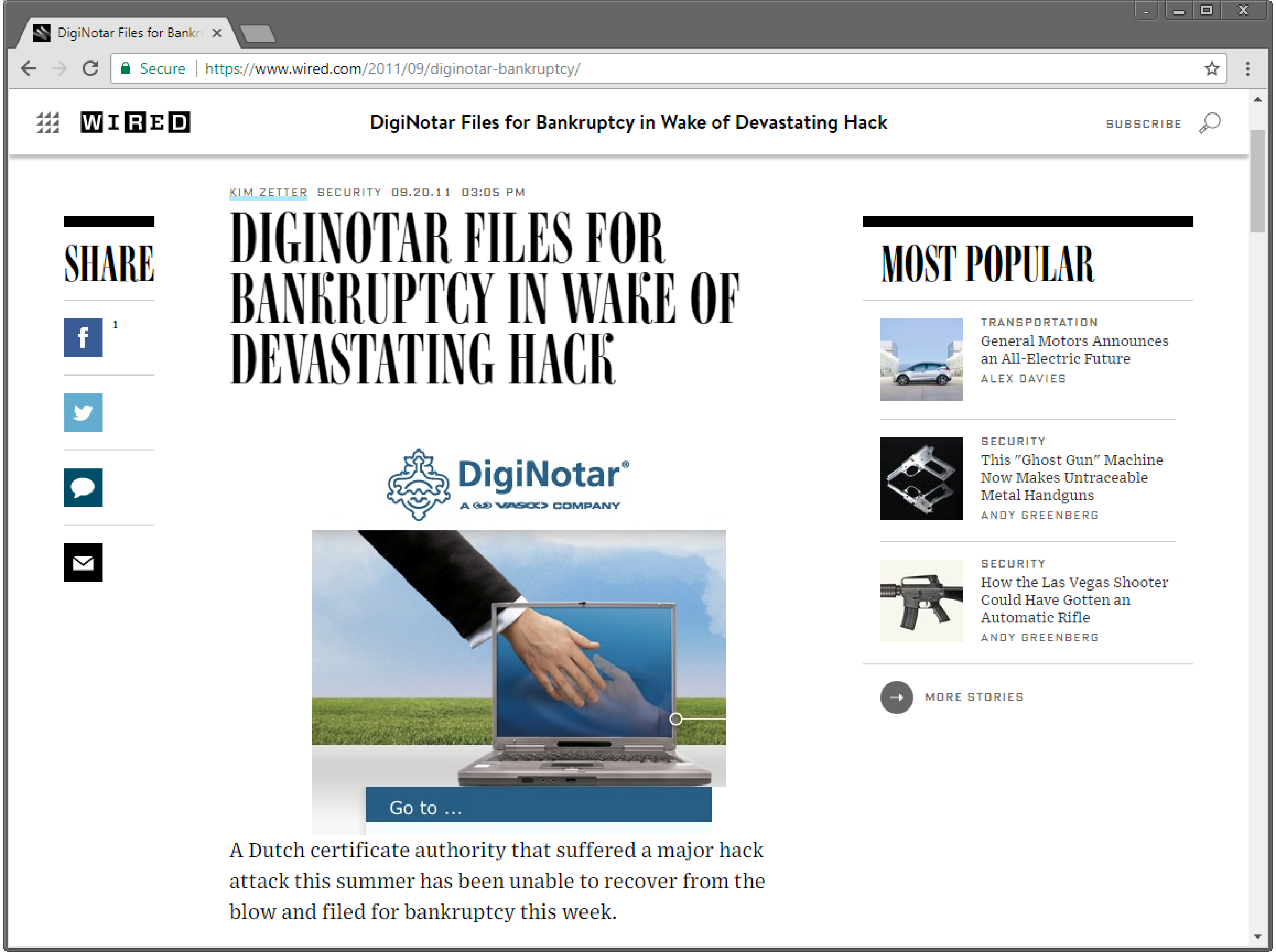
Allow revocation of compromised or no longer needed certificates

Certificate revocation list (CRL)

- Signed list of all revoked certificates that have not yet expired
- Main problem: lists tend to be large, making real-time lookups slow
- Can the attacker block connectivity to the CA's server?
- CRLSets (Chrome): revocation list pushed to the browser as a *software update*

Online Certificate Status Protocol (OCSP)

- Obtain the revocation status of a *single* certificate → faster
- But the latency, security, and privacy issues still remain
- OCSP stapling (Firefox): server embeds OCSP response directly into the TLS handshake (soft-fail issue remains: an adversary can suppress the OCSP response)



DigiNotar Files for Bankruptcy in Wake of Devastating Hack

KIM ZETTER SECURITY 09.20.11 03:05 PM

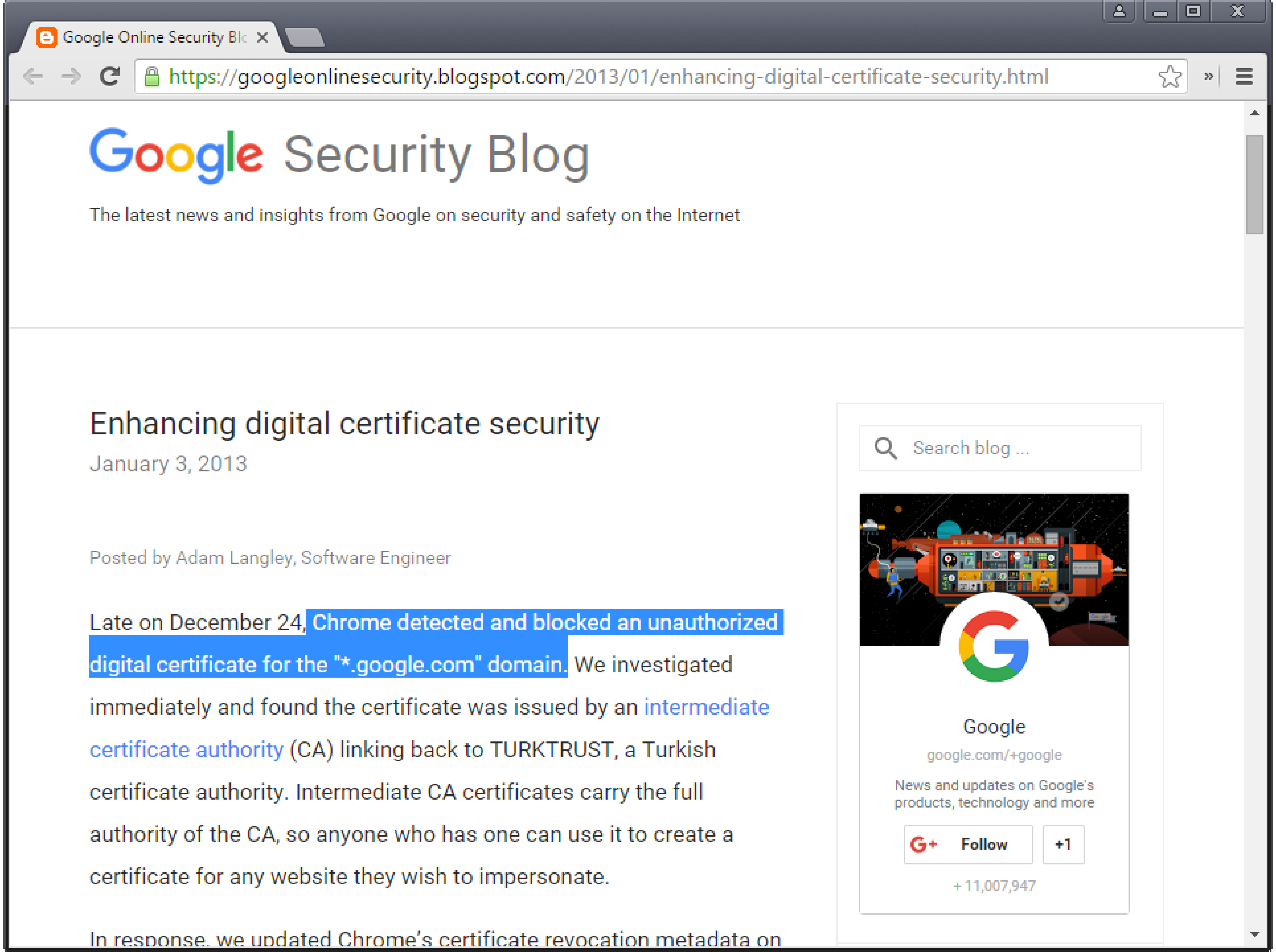
DIGINOTAR FILES FOR BANKRUPTCY IN WAKE OF DEVASTATING HACK

A Dutch certificate authority that suffered a major hack attack this summer has been unable to recover from the blow and filed for bankruptcy this week.

TRANSPORTATION
General Motors Announces an All-Electric Future
ALEX DAVIES

SECURITY
This "Ghost Gun" Machine Now Makes Untraceable Metal Handguns
ANDY GREENBERG

SECURITY
How the Las Vegas Shooter Could Have Gotten an Automatic Rifle
ANDY GREENBERG


Google Online Security Blc
https://googleonlinesecurity.blogspot.com/2013/01/enhancing-digital-certificate-security.html

Google Security Blog

The latest news and insights from Google on security and safety on the Internet

Enhancing digital certificate security

January 3, 2013

Posted by Adam Langley, Software Engineer

Late on December 24, Chrome detected and blocked an unauthorized digital certificate for the "*.google.com" domain. We investigated immediately and found the certificate was issued by an intermediate certificate authority (CA) linking back to TURKTRUST, a Turkish certificate authority. Intermediate CA certificates carry the full authority of the CA, so anyone who has one can use it to create a certificate for any website they wish to impersonate.

In response, we updated Chrome's certificate revocation metadata on

Certificate Transparency

Public monitoring and auditing of certificates

- Identify mistakenly or maliciously issued certificates and rogue CAs

Certificate logs

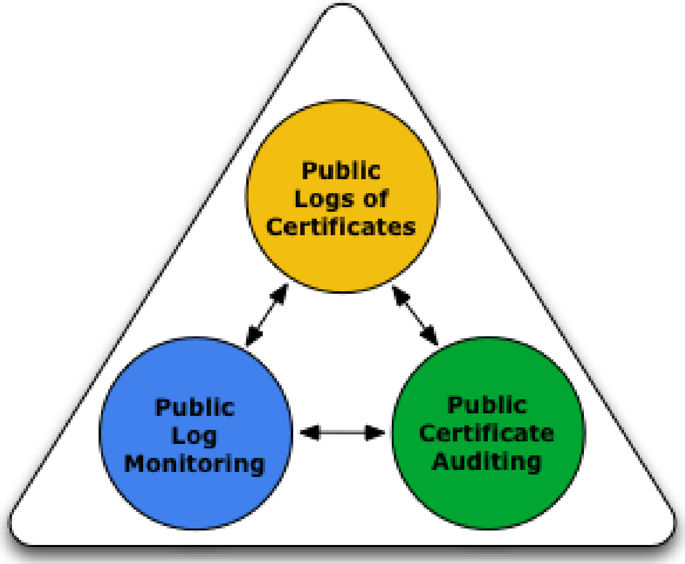
- Network services maintaining cryptographically assured, publicly auditable, append-only records of certificates

Monitors

- Periodically contact all log servers and watch for suspicious certificates

Auditors

- Verify that logs are behaving correctly and are cryptographically consistent
- Check that a particular certificate appears in a log



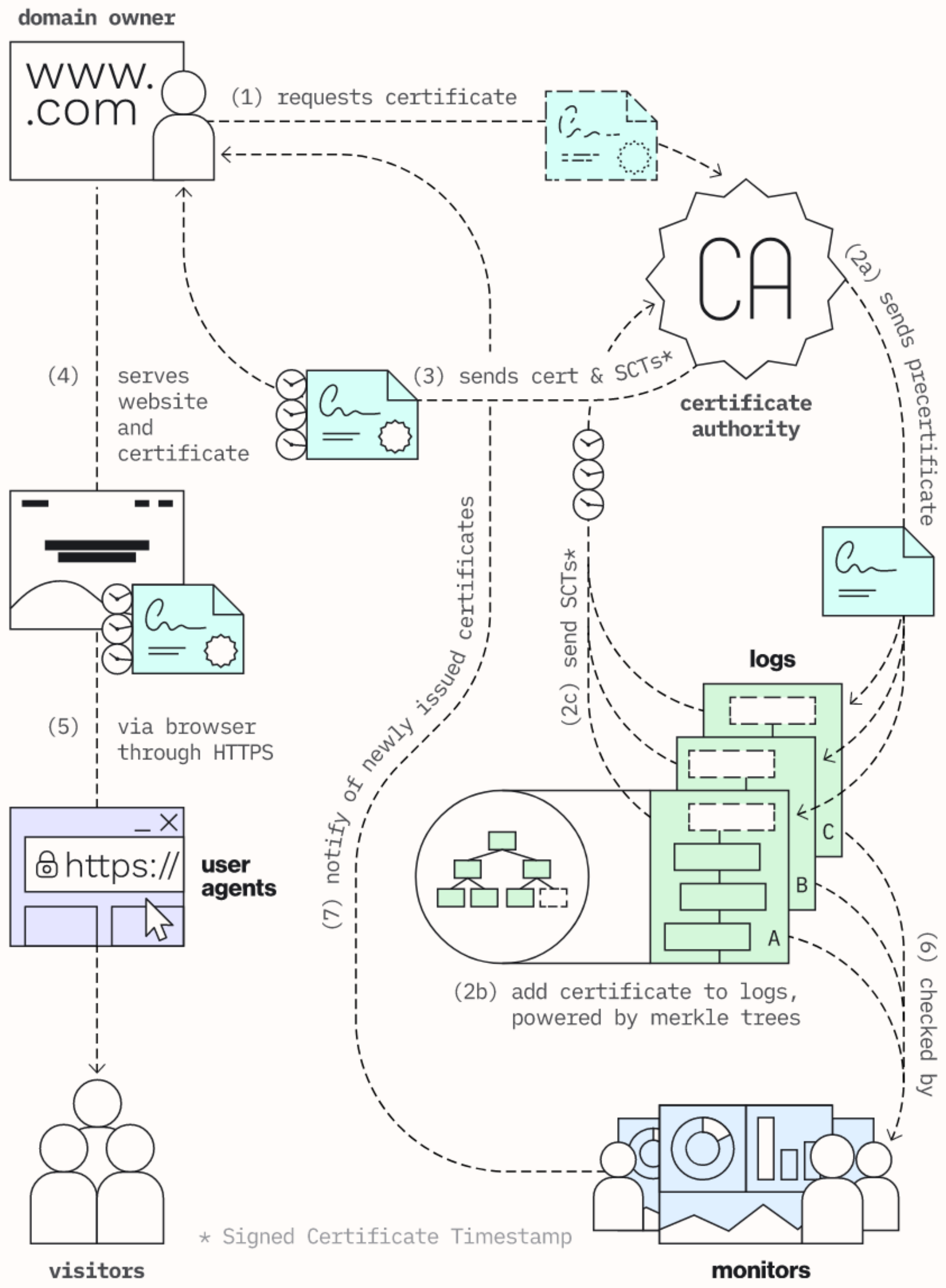
https://certificate.transparency.dev/

Certificates are deposited in public, transparent logs (append-only ledgers)

- Distributed and independent: anyone can query them to see what certificates have been included and when
- Append-only: verifiable by Monitors
- Web browsers enforce Certificate Transparency

Logs are cryptographically monitored

- Monitors cryptographically check which certificates have been included in logs
- Domain owners can subscribe to a CT monitor to get updates when precertificates/certificates for those domains are included in any of the logs checked by that monitor



domain owner
www.
.com
(1) requests certificate
CA
certificate authority
(2a) sends precertificate
(3) sends cert & SCTs*
(4) serves website and certificate
(2c) send SCTs*
(7) notify of newly issued certificates
logs
C
B
A
(2b) add certificate to logs, powered by merkle trees
(5) via browser through HTTPS
https://
user agents
(6) checked by
visitors
monitors

* Signed Certificate Timestamp

https://certificate.transparency.dev/howctworks/

Web of Trust (mainly used in PGP for encrypted email)

Entirely decentralized authentication

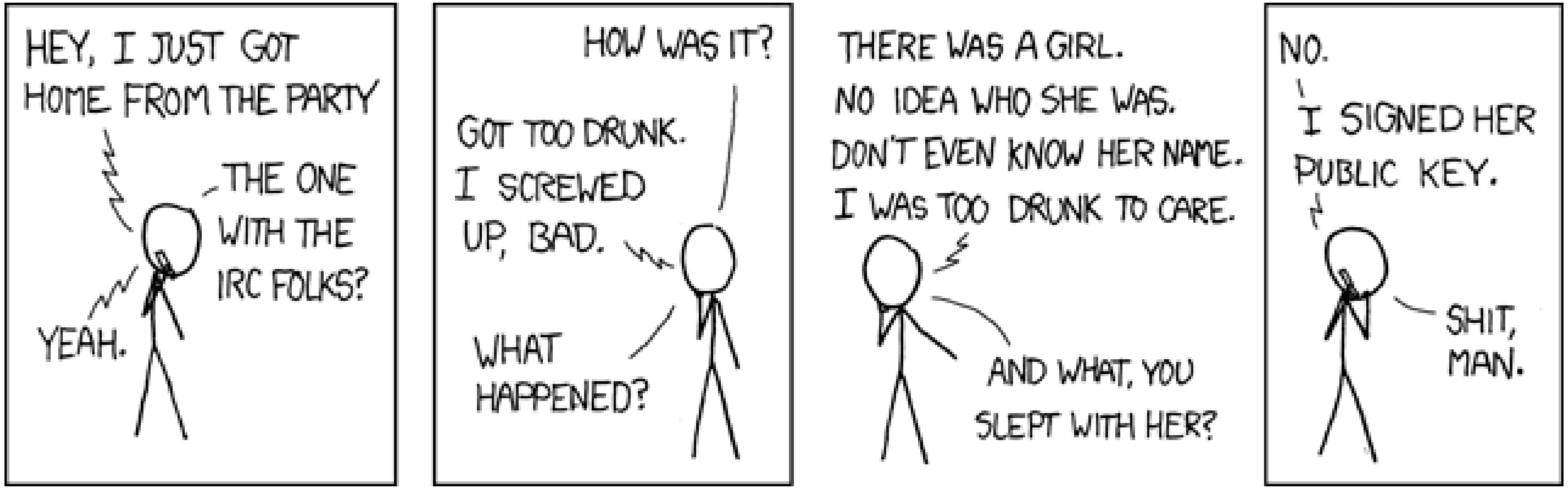
- No need to buy certs from CAs: users create their own certificates

Users validate other users' certificates, forming a "web of trust"

- No trusted authorities: trust is established through friends (yay! key signing parties!)
- Adjustable "skepticism" parameters: number of fully and partially trusted endorsers required to trust a new certificate (1 and 3 for GnuPG)

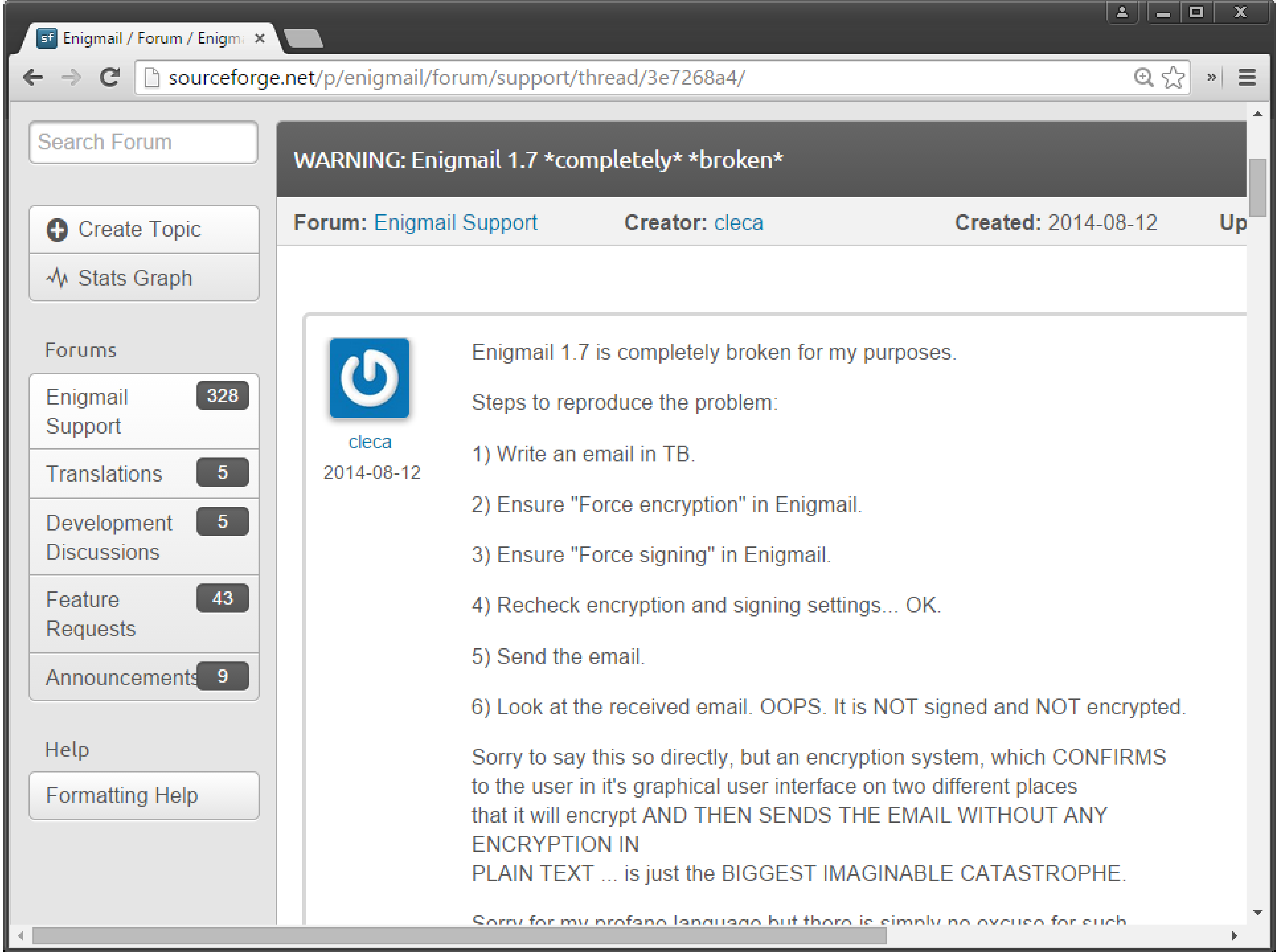
Main problems

- Privacy issues: social graph metadata
- Bootstrapping: new users are not readily trusted by others
- When opinions vary, "stronger set" wins: impersonation through collusion/compromised keys
- Scalability: WoT for the whole world?



HEY, I JUST GOT HOME FROM THE PARTY
THE ONE WITH THE IRC FOLKS?
YEAH.
HOW WAS IT?
GOT TOO DRUNK. I SCREWED UP, BAD.
WHAT HAPPENED?
THERE WAS A GIRL. NO IDEA WHO SHE WAS. DON'T EVEN KNOW HER NAME. I WAS TOO DRUNK TO CARE.
AND WHAT, YOU SLEPT WITH HER?
NO.
I SIGNED HER PUBLIC KEY.
SHIT, MAN.

https://xkcd.com/364/



WARNING: Enigmail 1.7 *completely* *broken*

Forum: Enigmail Support **Creator:** cleca **Created:** 2014-08-12 **Up**

cleca
2014-08-12

Enigmail 1.7 is completely broken for my purposes.

Steps to reproduce the problem:

1) Write an email in TB.

2) Ensure "Force encryption" in Enigmail.

3) Ensure "Force signing" in Enigmail.

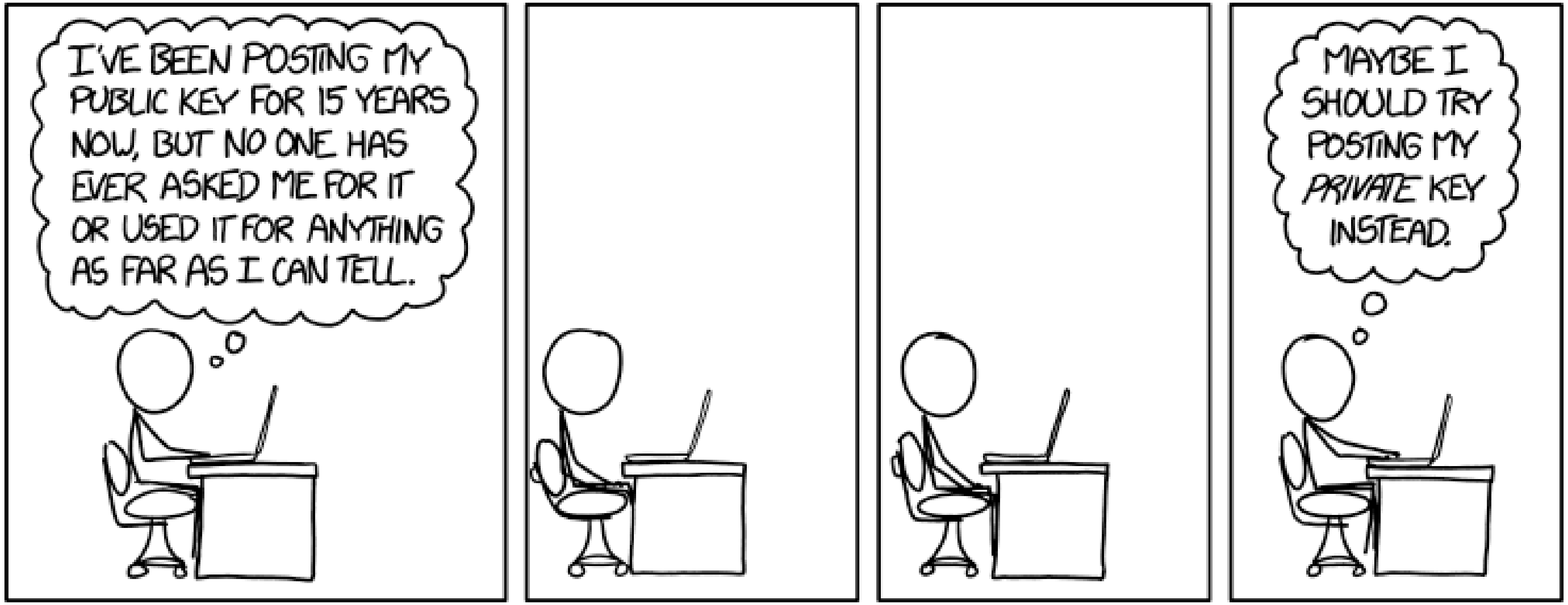
4) Recheck encryption and signing settings... OK.

5) Send the email.

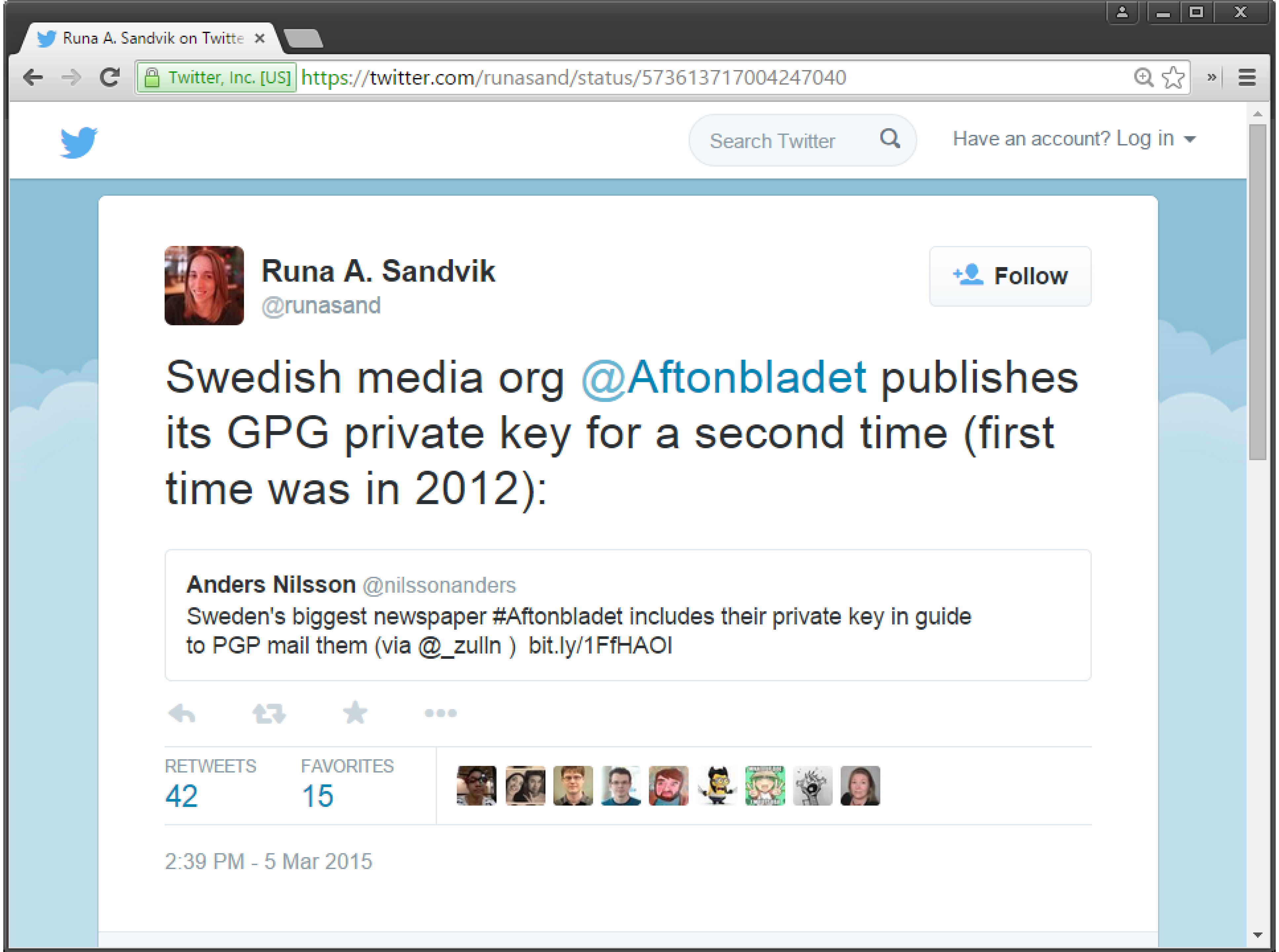
6) Look at the received email. OOPS. It is NOT signed and NOT encrypted.

Sorry to say this so directly, but an encryption system, which CONFIRMS
to the user in it's graphical user interface on two different places
that it will encrypt AND THEN SENDS THE EMAIL WITHOUT ANY
ENCRYPTION IN
PLAIN TEXT ... is just the BIGGEST IMAGINABLE CATASTROPHE.

Sorry for my profane language but there is simply no excuse for such


I'VE BEEN POSTING MY PUBLIC KEY FOR 15 YEARS NOW, BUT NO ONE HAS EVER ASKED ME FOR IT OR USED IT FOR ANYTHING AS FAR AS I CAN TELL.
MAYBE I SHOULD TRY POSTING MY *PRIVATE* KEY INSTEAD.

https://xkcd.com/1553/


Runa A. Sandvik on Twitte
Twitter, Inc. [US]
https://twitter.com/runasand/status/573613717004247040
Search Twitter
Have an account? Log in
Follow

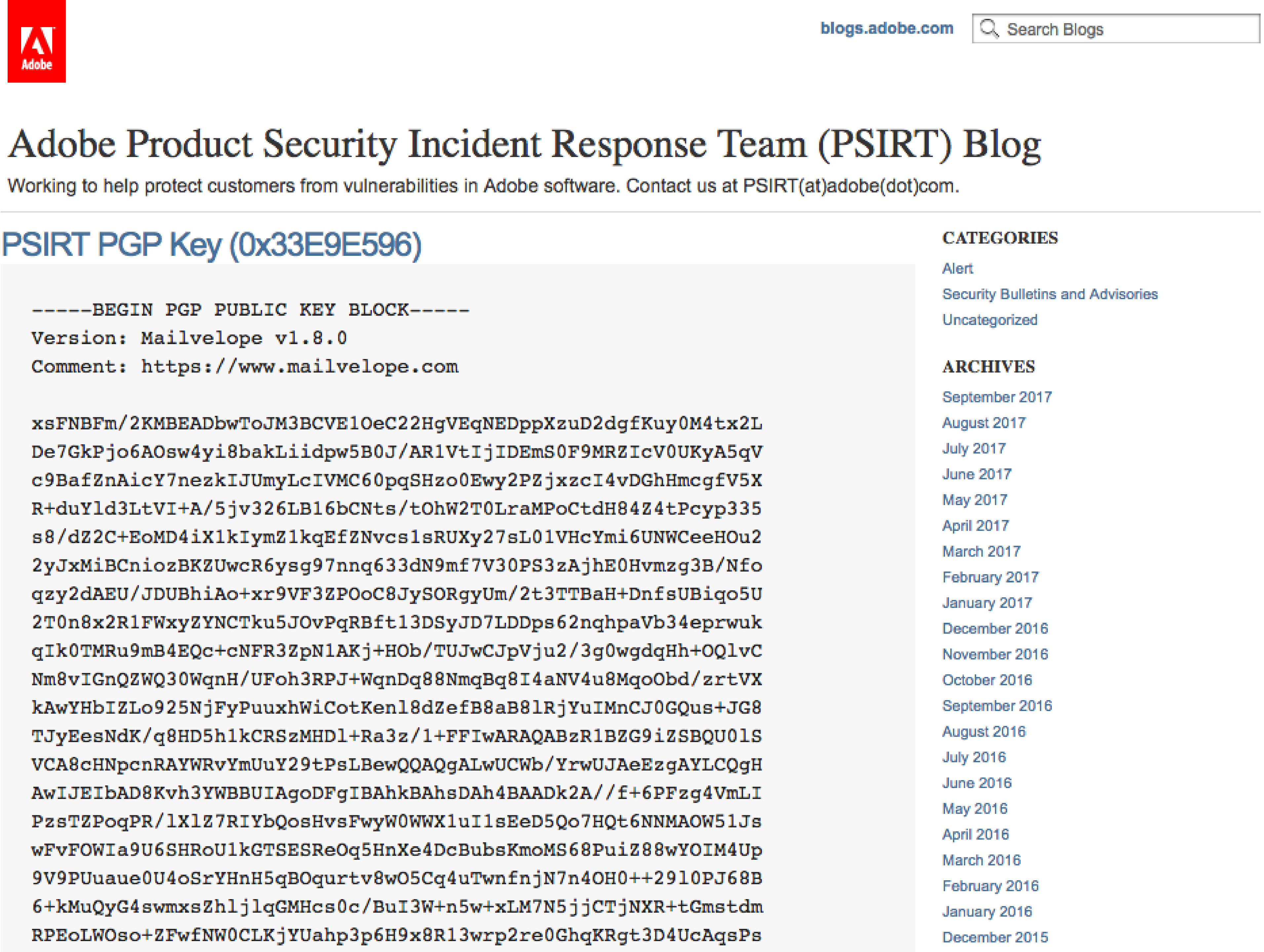
Runa A. Sandvik @runasand

Swedish media org @Aftonbladet publishes its GPG private key for a second time (first time was in 2012):

Anders Nilsson @nilssonanders
Sweden's biggest newspaper #Aftonbladet includes their private key in guide to PGP mail them (via @_zulln) bit.ly/1FfHAOI

RETWEETS	FAVORITES
42	15

2:39 PM - 5 Mar 2015



Adobe Product Security Incident Response Team (PSIRT) Blog

Working to help protect customers from vulnerabilities in Adobe software. Contact us at PSIRT(at)adobe(dot)com.

PSIRT PGP Key (0x33E9E596)

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: Mailvelope v1.8.0
Comment: https://www.mailvelope.com

xsFNBFm/2KMBEADbwToJM3BCVE1OeC22HgVEqNEDppXzuD2dgfKuy0M4tx2L
De7GkPjo6AOsw4yi8bakLiidpw5B0J/AR1VtIjIDEmS0F9MRZIcV0UKyA5qV
c9BafZnAicY7nezkIJUmyLcIVMC60pqSHzo0Ewy2PZjxzcI4vDGhHmcgfV5X
R+duYld3LtVI+A/5jv326LB16bCNts/tOhW2T0LraMPoCtdH84Z4tPcyp335
s8/dZ2C+EoMD4iX1kIymZ1kqEfZNvcs1sRUXy27sL01VHcYmi6UNWCeeHOu2
2yJxMiBCniozBKZUwcR6ysg97nnq633dN9mf7V30PS3zAjhE0Hvmzg3B/Nfo
qzy2dAEU/JDUBhiAo+xr9VF3ZPOoC8JySORgyUm/2t3TTBaH+DnfsUBiqo5U
2T0n8x2R1FWxyZYNCTku5JOvPqRBft13DSyJD7LDDps62nqhpaVb34eprwuk
qIk0TMRu9mB4EQc+cNFR3ZpN1AKj+HOb/TUJwCJpVju2/3g0wgdqHh+OQlvC
Nm8vIGnQZWQ30WqnH/UFoh3RPJ+WqnDq88NmqBq8I4aNV4u8MqoObd/zrtVX
kAwYHbIZLo925NjFyPuuxhWiCotKenl8dZefB8aB8lRjYuIMnCJ0GQus+JG8
TJyEesNdK/q8HD5h1kCRSzMHDl+Ra3z/1+FFIwARAQABzR1BZG9iZSBQU01S
VCA8cHNpcnRAYWRvYmUuY29tPsLBewQQAQgALwUCWb/YrwUJAeEzgAYLCQgH
AwIJEIbAD8Kvh3YWBBUIAgoDFgIBAhkBAhsDAh4BAADk2A//f+6PFzg4VmLI
PzsTZPoqPR/lX1Z7RIYbQosHvsFwyW0WWX1uI1sEeD5Qo7HQt6NNMAOW51Js
wFvFOWIa9U6SHRoU1kGTSESReOq5HnXe4DcBubsKmoMS68PuiZ88wYOIM4Up
9V9PUuaue0U4oSrYHnH5qBOqurtv8wO5Cq4uTwnfnjN7n4OH0++2910PJ68B
6+kMuQyG4swmxsZhljlqGMHcs0c/BuI3W+n5w+xLM7N5jjCTjNXR+tGmstdm
RPEoLWOso+ZFwfNW0CLKjYUahp3p6H9x8R13wrp2re0GhqKRgt3D4UcAqsPs
```

CATEGORIES

- Alert
- Security Bulletins and Advisories
- Uncategorized

ARCHIVES

- September 2017
- August 2017
- July 2017
- June 2017
- May 2017
- April 2017
- March 2017
- February 2017
- January 2017
- December 2016
- November 2016
- October 2016
- September 2016
- August 2016
- July 2016
- June 2016
- May 2016
- April 2016
- March 2016
- February 2016
- January 2016
- December 2015

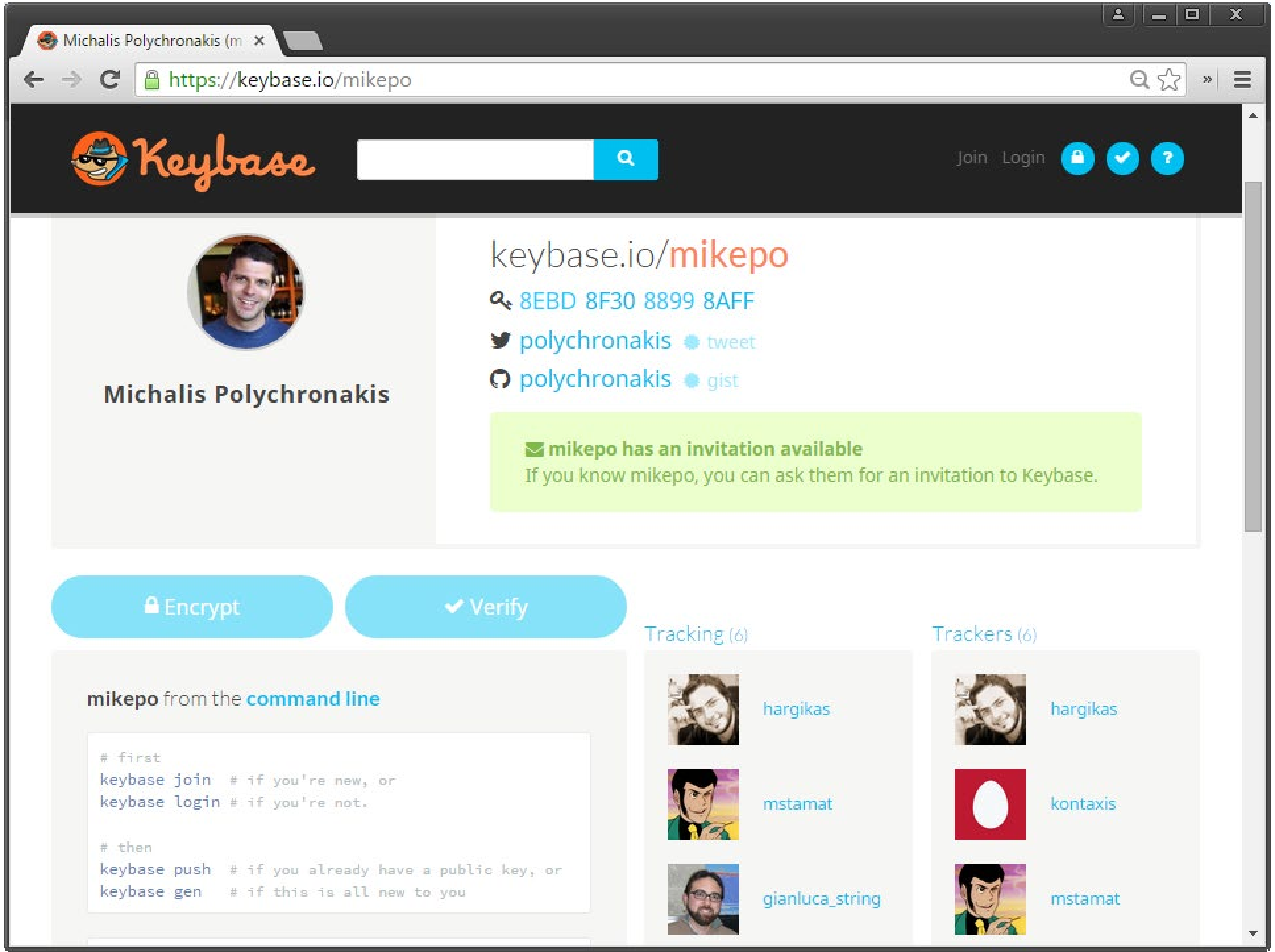
```
GAEIABkFAlm/2LAFCQHhM4AJEIbAD8Kvh3YWAhsMAACz+g/+KmbnChEUZXdo
ZIvPzphw3KvZQHWCY+5qGqdoxNkfkUSKhkzC0M51Kq7emVpvXYrMRdJRHxFP
83HIahA5UiufsDt7QlMwVRGtJYxhH+TNZBBbDBVQ1JQxuC3mH7F/tFHb9N1G
kURUwa2fdDBPw2+DOWa2+iVhcPhfB2iy9exs2txXjgPx67aZi70Jw44ixvpY
TWs/M5I6SXQsyuB5Qw0jtXKioQyTOLmeUFmJR2Ui5FK+t5SXus44mRCujEUn
YDqDmxKDnhssEVNWZ4KWs2uvNXNwlnZcHVSYXukf3FlCWp0TESCOecdqbvl0
Cs+vLivxiksh33xqZWnD78xv92t2Ggp2a41gBOaaCjx2irqZ9RHIv0YzNfQz
yz5XYEGI2iCrvdStrbZfX1Dqsllrqs/pZRbV48KbfubDvGZuNR3hrsfmfsgr
zkESOQmpuKhj/Es3CKjdafLDc8HOyVhJ+n4tvWXyRpYEhuDh/tzeDuuB9vfG
QA9TNhSpAp5lHFJklmd9knWbExJ0srUbK2QVmVn9CZx/sdUfwDWp1GeANLsO
MRNlr3IrklbZ0bFH+nrcJQZ5+sDzHGNe4P9Dt30yvFHoyS1BkRndLuawSlqh
LJyYLUvFjL3i3jbiNT1NKldwqaL2i9OuRAuHthoFGOKIqr6hmtOYzUem/cl+
ZlRwd77Vmfc=
=QOc7
-----END PGP PUBLIC KEY BLOCK-----

-----BEGIN PGP PRIVATE KEY BLOCK-----
Version: Mailvelope v1.8.0
Comment: https://www.mailvelope.com

xcaGBFm/2KMBEADbwToJM3BCVE1OeC22HgVEqNEDppXzuD2dgfKuy0M4tx2L
De7GkPjo6AOsw4yi8bakLiidpw5B0J/AR1VtIjIDEmS0F9MRZIcV0UKyA5qV
c9BafZnAicY7nezkIJUmyLcIVMC60pqSHzo0Ewy2PZjxzcI4vDGhHmcgfV5X
R+duYld3LtVI+A/5jv326LB16bCNts/tOhW2T0LraMPoCtdH84Z4tPcyp335
s8/dZ2C+EoMD4iX1kIymZ1kqEfZNvcs1sRUXy27sL01VHcYmi6UNWCeeHOu2
2yJxMiBCniozBKZUwcR6ysg97nnq633dN9mf7V30PS3zAjhE0Hvmzg3B/Nfo
qzy2dAEU/JDUBhiAo+xr9VF3ZPOoC8JySORgyUm/2t3TTBaH+DnfsUBiqo5U
2T0n8x2R1FWxyZYNCTku5JOvPqRBft13DSyJD7LDDps62nqhpaVb34eprwuk
qIk0TMRu9mB4EQc+cNFR3ZpN1AKj+HOb/TUJwCJpVju2/3g0wgdqHh+OQlvC
Nm8vIGnQZWQ30WqnH/UFoh3RPJ+WqnDq88NmqBq8I4aNV4u8MqoObd/zrtVX
```

April 2014
March 2014
February 2014
January 2014
December 2013
November 2013
October 2013
September 2013
July 2013
June 2013
May 2013
April 2013
March 2013
February 2013
January 2013
December 2012
November 2012
October 2012
September 2012
August 2012
June 2012
May 2012
April 2012
March 2012
February 2012
January 2012
December 2011
November 2011
October 2011
September 2011
August 2011
June 2011
May 2011
April 2011
March 2011
February 2011
December 2010

Finding Public Keys

- Public PGP key servers
  - pgp.mit.edu
  - keyserver.pgp.com
- Cache certificates from received emails
- Integration with user management systems (LDAP)
- Ad-hoc approaches
  - List public key on home page
  - Print on business card
  - Exchange through another medium on a case-by-case basis
- Association with social profiles/identities
  - keybase.io

WoT Alternative: Online Social “Tracking”



keybase.io/mikepo

8EBD 8F30 8899 8AFF

polychronakis tweet

polychronakis gist

Michalis Polychronakis

mikepo has an invitation available
If you know mikepo, you can ask them for an invitation to Keybase.

mikepo from the command line

```
# first
keybase join  # if you're new, or
keybase login # if you're not.

# then
keybase push  # if you already have a public key, or
keybase gen   # if this is all new to you
```

Tracking (6)	Trackers (6)
hargikas	hargikas
mstamat	kontaxis
gianluca_string	mstamat

Keybase.io

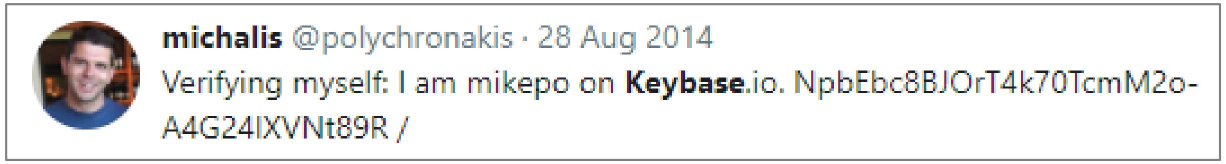
In essence, a directory associating public keys with names

Identity established through *public signatures*

- **Identity proofs**: *"I am Joe on Keybase and MrJoe on Twitter"*
- **Follower statements**: *"I am Joe on Keybase and I just looked at Chris's identity"*
- **Key ownership**: *"I am Joe on Keybase and here's my public key"*
- **Revocations**: *"I take back what I said earlier"*

Keybase identity = sum of public identities

- Twitter, Facebook, Github, Reddit, domain ownership, …



michalis @polychronakis · 28 Aug 2014

Verifying myself: I am mikepo on **Keybase**.io. NpbEbc8BJOrT4k70TcmM2o-A4G24IXVNt89R /

An attacker has to compromise all connected identities

- The more connected identities, the harder to impersonate a user

Best Practices

Use long passphrases instead of passwords

- **Never reuse the same password** on different services

Use two-factor authentication when available

- **Avoid SMS if possible!** Use an authenticator app or U2F instead
- Remove phone number from account after authenticator/U2F setup
- Store your backup codes in a safe location

Use a password manager

- Pick non-memorable passwords and avoid copy/pasting them
- **Password auto-fill helps against phishing!** (auto-fill will fail if the domain is wrong)

Use SSH keys instead of passwords