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
**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*: cryptography, secret tokens, location, …

Computers (in contrast to humans) can “remember” large secrets (keys or tokens) 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 relies on a trusted third party to establish authentication → *Identity and Access Management (IAM) services* (e.g., Okta, Duo, OneLogin)
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 $\Rightarrow$ easy to guess
- Hard to guess $\Rightarrow$ 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, …
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.
mikepo@styx:~> zxcvbn
Password:
```
{
  "password": "Tr0ub4dor&3",
  "guesses": "100000000001",
  "guesses_log10": 11.000000000004341,
  "crack_times_seconds": {
    "online_throttling_100_per_hour": "3600000000036.000199840144435",
    "online_no_throttling_10_per_second": "10000000000.1",
    "offline_slow_hashing_1e4_per_second": "1000000.0001",
    "offline_fast_hashing_1e10_per_second": "10.0000000001"
  }
}
```

### Complexity:
- **Score**: Very Strong
- **Additions**:
  - Number of characters: Flat, Rate: +n^4, Count: 11, Bonus: +44
  - Uppercase letters: Condf/ncr, Rate: +((len-n)^2), Count: 1, Bonus: +20
  - Lowercase letters: Condf/ncr, Rate: +((len-n)^2), Count: 6, Bonus: +10
  - Numbers: Condf, Rate: +n^4, Count: 3, Bonus: +12
  - Symbols: Flat, Rate: +n^6, Count: 1, Bonus: +6
  - Middle numbers or symbols: Flat, Rate: +n^2, Count: 3, Bonus: +6
  - Requirements: Flat, Rate: +n^2, Count: 5, Bonus: +10

### Deductions:
- Letters only: Flat, Rate: -n, Count: 0
- Numbers only: Flat, Rate: 0
- Repeat Characters (case insensitive): Comp, Rate: -2
- Consecutive uppercase letters: Flat, Rate: -(n^2), Count: 0
- Consecutive lowercase letters: Flat, Rate: -(n^2), Count: 3
- Consecutive numbers: Flat, Rate: -(n^3), Count: 0
- Sequential letters (3+): Flat, Rate: -(n^3), Count: 0
- Sequential numbers (3+): Flat, Rate: -(n^3), Count: 0
- Sequential symbols (3+): Flat, Rate: -(n^3), Count: 0

### Crack Times:
- Online throttling 100 per hour: centuries
- Online no throttling 10 per second: centuries
- Offline slow hashing 1e4 per second: 4 months
- Offline fast hashing 1e10 per second: 10 seconds
Password: correct horse battery staple

crack_times_seconds: {
    "online_throttling_100_per_hour": "7697230082272000427202.147568",
    "online_no_throttling_10_per_second": "213811968952000000000",
    "offline_slow_hashing_1e4_per_second": "2138119689520000000000",
    "offline_fast_hashing_1e10_per_second": "21381196895.2"
},
crack_times_display: {
    "online_throttling_100_per_hour": "centuries",
    "online_no_throttling_10_per_second": "centuries",
    "offline_slow_hashing_1e4_per_second": "centuries",
    "offline_fast_hashing_1e10_per_second": "centuries"
}

<table>
<thead>
<tr>
<th>Additions</th>
<th>Type</th>
<th>Rate</th>
<th>Count</th>
<th>Bonus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of characters</td>
<td>Flat</td>
<td>+n*(n^4)</td>
<td>28</td>
<td>+112</td>
</tr>
<tr>
<td>Uppercase letters</td>
<td>Cond/Incr</td>
<td>+((len-n)*(n)^2)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Lowercase Letters</td>
<td>Cond/Incr</td>
<td>+((len-n)*(n)^2)</td>
<td>25</td>
<td>+6</td>
</tr>
<tr>
<td>Numbers</td>
<td>Cond</td>
<td>+n*(n^4)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Symbols</td>
<td>Flat</td>
<td>+n*(n^6)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Middle numbers or symbols</td>
<td>Flat</td>
<td>+n*(n^2)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Requirements</td>
<td>Flat</td>
<td>+n*(n^2)</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deductions</th>
<th>Type</th>
<th>Rate</th>
<th>Count</th>
<th>Bonus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Letters only</td>
<td>Flat</td>
<td>-n</td>
<td>28</td>
<td>-28</td>
</tr>
<tr>
<td>Numbers only</td>
<td>Flat</td>
<td>-n</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Repeat Characters (case insensitive)</td>
<td>Cond</td>
<td>-</td>
<td>20</td>
<td>-2</td>
</tr>
<tr>
<td>Consecutive uppercase letters</td>
<td>Flat</td>
<td>-(n^2)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Consecutive lowercase letters</td>
<td>Flat</td>
<td>-(n^2)</td>
<td>24</td>
<td>-48</td>
</tr>
<tr>
<td>Consecutive numbers</td>
<td>Flat</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sequential letters (3+)</td>
<td>Flat</td>
<td>-(n^3)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sequential numbers (3+)</td>
<td>Flat</td>
<td>-(n^3)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sequential symbols (3+)</td>
<td>Flat</td>
<td>-(n^3)</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Password Policies (often have the opposite effect)

Password rules (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, and 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 ➔ ...
If the chosen secret is found in the list, the CSP or verifier SHALL advise the subscriber to choose a different secret and provide the reason for rejection, and SHALL require the subscriber to choose a different secret.

Verifiers SHOULD offer guidance to the subscriber, such as a password-strength tool or memorized secret. This is particularly important following the rejection of a memorized secret. 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).

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 SHALL not impose other composition rules (e.g., requiring mixtures of different character types or prohibiting consecutively repeated characters) for memorized secrets. Verifiers SHALL force a change if there is evidence of compromise of the authenticator.

Verifiers SHALL 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

Eavesdropping

Capturing

Brute force attacks
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, …)

Unique salt per user (no need to be secret): 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!

<table>
<thead>
<tr>
<th>Username</th>
<th>Salt</th>
<th>Password hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bobbie</td>
<td>4238</td>
<td>h(4238, $uperman)</td>
</tr>
<tr>
<td>Tony</td>
<td>2918</td>
<td>h(2918, 63%TaeFF)</td>
</tr>
<tr>
<td>Mitsos</td>
<td>6902</td>
<td>h(6902, zour1da)</td>
</tr>
<tr>
<td>Mark</td>
<td>1694</td>
<td>h(1694, Rockybrook#1)</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>Hash-Mode</th>
<th>Hash-Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>MD5</td>
<td>8743b52063cd94097a65d163f3c7f574f5</td>
</tr>
<tr>
<td>10</td>
<td>md5($pass,$salt)</td>
<td>01dfae6e5d4d90d98262325959a7be:7050461</td>
</tr>
<tr>
<td>20</td>
<td>md5($salt,$pass)</td>
<td>f0fda580370a6d9d1a7d80f0a4ceda2:4225637426</td>
</tr>
<tr>
<td>30</td>
<td>md5(sha1($pass),$salt)</td>
<td>b31d032efdc47a399999a71e43c5d2a:144816</td>
</tr>
<tr>
<td>40</td>
<td>md5($salt,sha1($pass))</td>
<td>d63d0e2f1f0c05f618d55e30654af82:13288442151473</td>
</tr>
<tr>
<td>50</td>
<td>HMAC-MD5 (key = $pass)</td>
<td>fc741b0b2896639dc2a50c78b03370:1234</td>
</tr>
<tr>
<td>60</td>
<td>HMAC-MD5 (key = $salt)</td>
<td>b7f280436f45a38eaacac3b00518f29:1234</td>
</tr>
<tr>
<td>100</td>
<td>SHA1</td>
<td>b89eac7e61417341b710b7277682940e6a277b</td>
</tr>
<tr>
<td>110</td>
<td>sha1($pass,$salt)</td>
<td>2f5a6647737e1b7b33239d43241e0d4073571:2014</td>
</tr>
<tr>
<td>120</td>
<td>sha1($salt,$pass)</td>
<td>cac35e2c06d96d7d7c0b655f31d94250b7506b2b:5363520024</td>
</tr>
<tr>
<td>130</td>
<td>sha1(sha1($pass),$salt)</td>
<td>c576f6ac5b71f145a07b089159fa47c7275bc87c2:631225</td>
</tr>
<tr>
<td>140</td>
<td>sha1(sha1($salt),$pass)</td>
<td>5d8e6d4877e6c7969cf0d2456ad39a4c67663a:8763434884872</td>
</tr>
<tr>
<td>150</td>
<td>HMAC-SHA1 (key = $pass)</td>
<td>ca9863a6f70f61bc3fb9a1be222aa860e5ea717:1234</td>
</tr>
<tr>
<td>160</td>
<td>HMAC-SHA1 (key = $salt)</td>
<td>e9b0244b01b5c39e46a28ca839a4b0c3aaeaae1:1234</td>
</tr>
<tr>
<td>200</td>
<td>MySQL22</td>
<td>71967592100d60f0</td>
</tr>
<tr>
<td>300</td>
<td>MySQL4.1/MySQL5</td>
<td>fc7f1c186749d95d8e5f34271d635178b5d130</td>
</tr>
</tbody>
</table>
### 50 Most-used (Worse) Passwords

<table>
<thead>
<tr>
<th>Password1</th>
<th>Password2</th>
<th>Password3</th>
<th>Password4</th>
<th>Password5</th>
<th>Password6</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456</td>
<td>1234567</td>
<td>123</td>
<td>ashley</td>
<td>ashley</td>
<td>evite</td>
</tr>
<tr>
<td>123456789</td>
<td>qwerty</td>
<td>omgpop</td>
<td>987654321</td>
<td>123abc</td>
<td>123qwe</td>
</tr>
<tr>
<td>picture1</td>
<td>abc123</td>
<td>123321</td>
<td>unknown</td>
<td>unknown</td>
<td>sunshine</td>
</tr>
<tr>
<td>password</td>
<td>Million2</td>
<td>654321</td>
<td>zxcvbnm</td>
<td>zxcvbnm</td>
<td>121212</td>
</tr>
<tr>
<td>12345678</td>
<td>000000</td>
<td>qwertyuiop</td>
<td>112233</td>
<td>chatbooks</td>
<td>1q2w3e4r</td>
</tr>
<tr>
<td>111111</td>
<td>1234</td>
<td>qwer123456</td>
<td>20100728</td>
<td>123456a</td>
<td>dragon</td>
</tr>
<tr>
<td>123123</td>
<td>iloveyou</td>
<td>123456a</td>
<td>123123123</td>
<td>123456</td>
<td>5201314</td>
</tr>
<tr>
<td>12345</td>
<td>aaron431</td>
<td>a123456</td>
<td>123123123</td>
<td>123456</td>
<td>159753</td>
</tr>
<tr>
<td>1234567890</td>
<td>password1</td>
<td>6666666</td>
<td>princess</td>
<td>jacket025</td>
<td>0123456789</td>
</tr>
<tr>
<td>senha</td>
<td>qqww1122</td>
<td>asdfghjkl</td>
<td>jacket025</td>
<td>jacket025</td>
<td>0123456789</td>
</tr>
</tbody>
</table>

[https://en.wikipedia.org/wiki/List_of_the_most_common_passwords](https://en.wikipedia.org/wiki/List_of_the_most_common_passwords)
Distribution of 4-digit sequences within RockYou passwords
ce#ebc.dk
goddess5
20071002
271075711
zs3cu72a
scoopn
frygas1411
SL12345651
12345687ee123
xuexi2010
daigoro
12345614
DICK4080
567891234
tilg80
6208c661
:zark:
ravishneha
15057161369
661189
passme
trolovinasveta
abdulkhalque
007816
xLD5x
Florida2011
037037
WestCountry
hitsugaiya
955998126
3n3rmx
4637324
bugger825
marmares
jinxin11
170383gp
3484427
f133321
zwqrgf
67070857
432106969
6856
704870704870
pv041886
20060814
512881535
milanimilani
472619
dbyxw888
85717221
cc841215
ariana19321
bbbnnpn
ang34hehiu
wj112358
Brenda85
786525pb
shia41988
pinyin
eybozip
71477nak
stokurew
ga8mw4yz
kukumbike
268888
jordi10
lexusis
kj01a039
c84bwlrb
prlyanka05
loveneverdies
u8aqeb576
FGYfg77
659397
327296
74748585
19720919
050769585
nicopa
2232566
bearss
n0public
isitreal00
ashraf19760
48144
22471015
antzyhou115
0167005246
ec11kag
226226226226
6767537/33
mimilebrock
gueis8850
fujinshan
counter
N8mr0n
520057
adc123
bmaster
cpbj642g
uelada77
EMANUELLI
yanjing
assyst
62157173
0704224950753
6903293
axaxa
hilal1
30091983
2510618981
soukuokpan
tosecondlife
p4os86mq
015614117
acw71790
lisyjmi2
2xgialdl
gaybar9
88203009
MKtyh87
quiggle
2063775206
fr3ih3it
masich
pengalwei
coalesce
56402768
thesis
aabbcc894
marlonmaxime
614850
yd220105
584521584521
txudecp
84410545
pietro.chiera
jman1514
heryarma
39joinmam
timelapse
mwinkar
251422
willrock
YHrftfdK
xys96exq
mercadotecnia
8s5sBE6x
0125043044
margitka
omaopa
dfti6nh
1314520521
pixa7m66
pearpear
gothpunksk8er
rtfae048
8d7R0K
5172032
aics07
304dmariah
dongqinwei
samarica
cap1014
0167387943
AE86Trueno
19700913
mcsuap
bu56mpbu
danbee
passw<
money521
conan83
nxjfp1
ragte143
kojiyhen
058336257
sarahr444
7363437
freindship
JytmvWO848
sb_inbau
30907891
001104
desare11
412724198
nhibikab
asferg
hqbp555
xgames7
muckerlee
choqui67
12130911
lierwei120
skytdvn
milenar1995
kambala11
# LEAKED LISTS

Complete left lists from public leaks

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Last Update</th>
<th>Num of Hashes</th>
<th>Progress</th>
<th>Left Hashes</th>
<th>Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>6505</td>
<td>H4v3 l b33n pwn3d (SHA1)</td>
<td>02.10.2017 - 02:03:24</td>
<td>320'294'464</td>
<td>319'837'535 (99.86%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>5638</td>
<td>P4y4sUGym (MD5)</td>
<td>02.10.2017 - 02:04:19</td>
<td>241'266</td>
<td>221'152 (91.66%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>4920</td>
<td>L1nk3d1n (SHA1)</td>
<td>02.10.2017 - 03:25:32</td>
<td>61'829'262</td>
<td>60'147'825 (97.28%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>3282</td>
<td>4mzr3v13w7r4d3r.c0m (MYSQL5)</td>
<td>02.10.2017 - 03:25:32</td>
<td>41'823</td>
<td>39'166 (93.65%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>3186</td>
<td>X5p1l7 (SHA1)</td>
<td>02.10.2017 - 03:32:38</td>
<td>2'227'254</td>
<td>2'162'101 (97.07%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>2499</td>
<td>Hashkiller 32-hex left total</td>
<td>02.10.2017 - 11:48:14</td>
<td>9'976'651</td>
<td>1'723'709 (17.28%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>2498</td>
<td>Hashkiller 40-hex left total</td>
<td>02.10.2017 - 13:22:34</td>
<td>1'739'204</td>
<td>350'788 (20.17%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>1619</td>
<td>4m5t3urc0mmuni7y.c0m</td>
<td>02.10.2017 - 13:33:26</td>
<td>197'302</td>
<td>57'407 (29.1%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>1535</td>
<td>b73r.c0m (MDS)</td>
<td>02.10.2017 - 13:34:43</td>
<td>63'070</td>
<td>32'543 (51.6%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>1427</td>
<td>4v17r0n.fr</td>
<td>02.10.2017 - 13:34:43</td>
<td>2'405</td>
<td>2'334 (97.05%)</td>
<td>Get</td>
<td>Get</td>
</tr>
<tr>
<td>1366</td>
<td>v0d4f0n34 (MDS($pass.<em>s+.a</em>))</td>
<td>02.10.2017 - 13:34:44</td>
<td>322</td>
<td>307 (95.34%)</td>
<td>Get</td>
<td>Get</td>
</tr>
</tbody>
</table>
## Largest Breaches

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

## Recently Added Breaches

<table>
<thead>
<tr>
<th>Count</th>
<th>Website Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>49,102,176</td>
<td>Alleged AT&amp;T accounts</td>
</tr>
<tr>
<td>3,262,980</td>
<td>ClickASnap accounts</td>
</tr>
<tr>
<td>552,094</td>
<td>Flipkart accounts</td>
</tr>
<tr>
<td>3,517,679</td>
<td>Habib's accounts</td>
</tr>
<tr>
<td>2,451,197</td>
<td>APK.TW accounts</td>
</tr>
<tr>
<td>3,805,265</td>
<td>Online Trade (Онлайн Трейд) accounts</td>
</tr>
<tr>
<td>21,994</td>
<td>WoTLabs accounts</td>
</tr>
<tr>
<td>27,123</td>
<td>Mr. Green Gaming accounts</td>
</tr>
<tr>
<td>19,972,829</td>
<td>Cutout.Pro accounts</td>
</tr>
<tr>
<td>243,462</td>
<td>Tangerine accounts</td>
</tr>
</tbody>
</table>
Password Hashing Functions

Hash functions are very fast to evaluate \(\Rightarrow\) 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)
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 stuffling*: 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 \(\Rightarrow\) attackers may now be able to lock you out

Better: disable password authentication altogether and use a key pair \(\Rightarrow\) cumbersome if having to log in from several devices or others’ computers
<table>
<thead>
<tr>
<th>LOGIN: mitch</th>
<th>LOGIN: carol</th>
<th>LOGIN: carol</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASSWORD: FooBar!-7</td>
<td>INVALID LOGIN NAME</td>
<td>PASSWORD: Idunno</td>
</tr>
<tr>
<td>SUCCESSFUL LOGIN</td>
<td>LOGIN:</td>
<td>INVALID LOGIN</td>
</tr>
<tr>
<td>(a)</td>
<td>(b)</td>
<td>LOGIN:</td>
</tr>
</tbody>
</table>

(a) Successful login  
(b) Login rejected after name is entered  
(c) Login rejected after name and password are typed  
   ➔ less information makes guessing harder
Before guessing, try the default first…

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Model</th>
<th>Protocol</th>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>CISCO</td>
<td>CACHE ENGINE</td>
<td>CONSOLE</td>
<td>admin</td>
<td>diamond</td>
</tr>
<tr>
<td>CISCO</td>
<td>CONFIGMAKER</td>
<td></td>
<td>cmaker</td>
<td>cmaker</td>
</tr>
<tr>
<td>CISCO</td>
<td>CNR Rev. ALL</td>
<td>CNR GUI</td>
<td>admin</td>
<td>changeme</td>
</tr>
<tr>
<td>CISCO</td>
<td>NETRANGER/SECURE IDS</td>
<td>MULTI</td>
<td>netrangi</td>
<td>attack</td>
</tr>
<tr>
<td>CISCO</td>
<td>BBSM Rev. 5.0 AND 5.1</td>
<td>TELNET OR NAMED PIPES</td>
<td>bbsd-client</td>
<td>changeme2</td>
</tr>
<tr>
<td>CISCO</td>
<td>BBSD MSDE CLIENT Rev. 5.0 AND 5.1</td>
<td>TELNET OR NAMED PIPES</td>
<td>bbsd-client</td>
<td>NULL</td>
</tr>
</tbody>
</table>
Eavesdropping and Replay

Physical world
- Post-it notes, notebooks, ...
- Lift fingerprints (e.g., Apple Touch ID)

Network
- 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, …
Password Capture
Hardware bugs/keyloggers
Software keyloggers/malware
Shoulder surfing
Cameras (e.g., ATM skimmers)
Social engineering
Press Ctrl-Alt-Delete to begin.

Requiring this key combination at startup helps keep computer secure. For more information, click Help.
(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, …
“The probability that a random person the population [sic] could look at your iPhone X and unlock it using Face ID is approximately 1 in 1,000,000 (versus 1 in 50,000 for Touch ID).

For additional protection, Face ID allows only five unsuccessful match attempts before a passcode is required to obtain access to your iPhone.

The probability of a false match is different for twins and siblings that look like you as well as among children under the age of 13, because their distinct facial features may not have fully developed. If you're concerned about this, we recommend using a passcode to authenticate.”

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 now 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 web session after authentication has been completed (malware, e.g., banking trojans)

Dual infection: compromise both PC and mobile device (rare)

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)

Social engineering

Call victim’s mobile operator and hijack the phone number
SIM swapping, 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
‘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.
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

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...
Hackers access some customer data at FTX, Genesis, and BlockFi by SIM swapping an employee of Krill, which manages creditor claims for the bankrupt companies — A cybersecurity incident: alleged fraud, which manages creditor claims of of bankrupt companies — Article Now
Aug 23, 2023, 7:06 PM — In-context

Isha Maruti / Forbes
Blockchain Capital co-founder Bart Stephens says a hacker who stole $6.3m in crypto via a SIM-swap attack: FBI: $72m was stolen via SIM swaps in 2022, up 6% YoY — Bart Stephens, co-founder and managing partner of crypto capital Blockchain Capital who was on early and prominent evangelist for cryptocurrencies — Article Now
Aug 23, 2023, 1:47 PM — In-context

Emma Roth / The Verge
CISA releases a report detailing LuaSoft's key techniques, calls for passwordless logins, and asks the FTC and the FCC for stricter SIM swapping protections — The US Cybersecurity and Infrastructure Security Agency (CISA) is calling for action on SIM swapping protections and the transition — Article Now
Aug 19, 2023, 12:08 PM — In-context

Blumberg:
How members of the Community, a group of teenage SIM swappers who met on the forum OGUsers, stole millions in crypto in 2018 before turning on each other — Members of the group were once considered to be a phone "SIM swap". When their war to get it back, they found themselves alone — Article Now
Aug 5, 2023, 9:46 AM — In-context

David Capretta / The Next Web
Federal judge refuses to dismiss $224M lawsuit against AT&T for allegedly letting a customer be SIM-swapped twice, leading to the loss of $2.6M in cryptocurrency — This investor lost 2.6M in cryptocurrency after being SIM-swapped twice — A federal judge has rejected AT&T's motion — Article Now
Jul 21, 2023, 12:36 PM — In-context

Andy Greenberg / Wired
While many foreign phone carriers are sharing real-time SIM swap data with banks to stop financial fraud, US carriers are dragging their feet — A 2018-2019 study found that the USA has 3.5 times the number of SIM swaps as the rest of the world combined — Article Now
Aug 27, 2023, 12:19 PM — In-context

Joseph Cox / Vice
A look at so-called Russian, encrypted, or "white" SIMs, used by criminals to spoof phone numbers, add voice manipulation to calls in real-time, and more — Criminals use so-called Russian, encrypted, or white SIMs to change their phone numbers, add voice manipulation to their calls — Article Now
Aug 12, 2023, 11:54 AM — In-context

Cristina Ciupi / Dietvorst:
Europal, working with US, UK, and others, says 10 people have been arrested for allegedly stealing $100M in cryptocurrency from celebrities via SIM-swap attacks — Eightteen were arrested in England and Scotland as part of an investigation into a series of SIM-swapping attacks targeting US celebrities. — Article Now
Aug 12, 2023, 2:40 PM — In-context

Stefan Marnot / Bloomberg:
Canada arrests a teenager for allegedly stealing $6.3M in crypto from a US citizen using SIM swapping, the largest reported single-person crypto theft — Canada marks the biggest crypto theft reported by one person — Police identified the alleged thief through a gaming session — Article Now
Nov 18, 2023, 10:04 AM — In-context

Lorenzo Franceschini / Bloomberg:
Europal, alongside Italian and Spanish police, arrest 16 people accused of working for the Italian Mafia and laundering over €10M made through cybercrimes — European police arrest several people of SIM swapping, phishing, and hacking in support of Italian-organized crime — Article Now
Sep 28, 2023, 11:08 AM — In-context

Results 1 - 10 of about 477:
Ashleigh Belanger / Ars Technica:
The US indicts a Chicago man who allegedly led a SIM-swap gang: members stole millions and posed as other people in Apple, T-Mobile, AT&T, and Verizon stores — Scheme allegedly targeted Apple, AT&T, Verizon, and Sprint — Two Federal Judges in Illinois — The US may have uncovered the hacker's identity — Article Now
Jan 30, 2024, 6:55 PM — In-context

MacKenzie Sigalas / CNBC:
The US SEC says the January 9 hack of its X account was via a SIM swap attack to reset its password; it had disabled 2FA in July 2023 over account access issues — The US Securities and Exchange Commission said on Wednesday that a SIM swap attack was used to hack the firm's X social media account — Article Now
Jan 22, 2023, 11:55 PM — In-context

Gary Miller / The Citizen Lab:
Research details how vulnerabilities in signaling protocols used by mobile network operators for international roaming can be exploited to move data through SIM cards — A detailed look at SIM swapping, a complex form of mobile phone fraud often used to steal cryptocurrencies and other forms of value — An investigation reveals how SIM swapping works. — Article Now
Oct 29, 2023, 2:23 PM — In-context

Lorenzo Franceschini / Bloomberg:
A college student who stole SIMs in cryptocurrency via SIM hijacking gets 10 years in prison and is the first person in the US to be sentenced for such a crime. — A 20-year-old college student has been found guilty of stealing more than $1 million in cryptocurrency in a case of SIM hijacking attack. — Article Now
Feb 5, 2023, 7:15 PM — In-context

Brian Krebs / Krebs on Security:
A detailed look at SIM swapping, a complex form of mobile phone fraud often used to steal cryptocurrencies and other forms of value — An investigation reveals how SIM swapping works. — Article Now
Nov 18, 2023, 8:08 AM — In-context

Researcher: AT&T, T-Mobile, Tracfone, US Mobile, and Verizon use vulnerable procedures for customer support that put users at risk of SIM swapping attacks — Researchers said that up to 1 in 5 major online services are vulnerable to SIM swapping attacks. — Article Now
Jan 11, 2023, 12:08 PM — In-context

New York Times:
Profile of Twitter hack mastermind Graham Clark, a troubled teen who allegedly went from scamming on Microsoft to joining hacking forums OGUsers and SIM swapping — The teenager "masterminded" the recent Twitter hack, which had a difficult family life, Posted his own leaks, and ran a new video game and cryptocurrency. — Article Now
Aug 5, 2023, 9:11 PM — In-context

Brian Krebs / Krebs on Security:
The Twitter attack may have been perpetrated by Joseph James Connor, a 21-year-old English SIM swapping linked to a group that hijacked Jack's account last year — Twitter was given three hours on Wednesday after accounts for some of the world's most high-profile public figures. — Article Now
Jul 14, 2023, 7:08 PM — In-context

Sergio Galler / Bloomberg:
T-Mobile confirms reports of a data breach caused by SIM swap attacks on a "very small number of customers", following six other data breaches since 2019. — T-Mobile confirmed reports of a new data breach and linked to notifications sent to "a very small number of customers" who fell victim to SIM swap attacks. — Article Now
Dec 23, 2023, 11:30 PM — In-context

Brian Krebs / Krebs on Security:
The SIM-simulating attacks are mostly claimed multiple times on Telegram to have phishing links to T-Mobile throughout 2022, far more than other US carriers. — There are different cybercriminals groups claiming access to internal networks of communications giant T-Mobile in more than 100... — Article Now
Feb 23, 2023, 5:02 PM — In-context
SMS as 2nd Factor vs. SMS for Account Recovery

Despite its shortcomings, SMS as a 2nd factor is better than nothing

Data point (Google): prevented 100% of 3.3B automated password stuffing attacks, 96% of 12M bulk phishing, and even 76% of <10k targeted attacks seen over a year

Unfortunately, the convenience of phone numbers has led many services to overload SMS as the sole authentication factor

- SMS-based onboarding
- SMS-based authentication (login with phone number)
- SMS-based password reset/account recovery

These are disastrous: a simple SIM-swap attack can take over an account without knowing the password

Password reset via email is much more secure
Better Alternative: Authenticator Apps

Time-based one-time password (TOTP)
- Six/eight digit code provided after password validation
- HMAC of a shared secret key and the current time
- The key is negotiated during registration

Requires “rough” client–server synchronization
- Code constantly changes in 30-second intervals

User-friendly alternative: push notification (e.g., Duo Push)
- MFA “fatigue” attacks: flood a user with push notifications

More importantly: **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 tricked into allowing device access due to overload of push notifications

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

Social engineering technique confuses victims to gain entry to their accounts

Malicious hackers are targeting Office 365 users with a spate 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
The attacker then repeatedly tried to log in to the Uber account using the illegally obtained credentials, prompting a two-factor login approval request each time. After the contractor initially blocked those requests, the attacker contacted the target on WhatsApp posing as tech support, telling the person to accept the MFA prompt — thus allowing the attacker to log in.
Evilginx2  [https://github.com/kgretzky/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)
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
Key Generation

Storing a private key + metadata per service would require a lot of storage

Alternative: store only a master symmetric key

Generated on-device upon first startup, and never leaves the YubiKey in any form

Registration

YubiKey generates a random key pair per credential

YubiKey encrypts the private key + metadata with the master key \(\rightarrow\) key handle

Key handle + public key sent to server

Authentication

The server presents the key handle to the YubiKey, along with a challenge

YubiKey decrypts the key handle and reveals the private key (authenticated encryption: ensures integrity, and that the credential is used with the correct AppID)

YubiKey signs the challenge with the private key to complete the authentication

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 ➔ prevents 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)
Google's strongest security helps keep your private information safe.

The Advanced Protection Program safeguards users with high visibility and sensitive information from targeted online attacks. New protections are automatically added to defend against today's wide range of threats.

Learn how to get started
First, you need 2 security keys, one of them for backup. Your security key will be used in addition to your password to sign in to your account. You can use keys that you already own or buy new ones. Learn more

Titan Security Key
From Google
Make sure to get key types that are compatible with your devices.

Buy now

Register security keys
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 another Apple device that supports FIDO2.
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 account recovery, 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 locally with a master password never visible to the cloud service

Single point of failure (!)
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

Use a central authentication service for multiple sites

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
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/
Passkeys

Completely replace passwords with cryptographic key pairs

Server only keeps a user’s public key

Based on WebAuthn: rely on biometric identification (Face ID, Windows Hello, …)

Key enabler: identity providers (Apple, Google, …) who also sell devices

The device becomes an authenticator: what if it gets lost? → recovery through vendor

Users have more than one device → seamless syncing

Sign in or sign up

Email
Enter email

Create account?

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

Sign up

Sign in

Do you want to save a passkey?

Continue with Touch ID
Save on another device
Passwordless by default: Make the switch to passkeys

For Cybersecurity Awareness Month we’re making it even easier for users to get started with passkeys.

Sriram Karra
Senior Product Manager

Christiaan Brand
Group Product Manager
## Passkeys.directory

Passkeys.directory is a community-driven index of websites, apps, and services that offer signing in with passkeys.

### Passkeys supported

<table>
<thead>
<tr>
<th>Name</th>
<th>Supported</th>
<th>Category</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adobe</td>
<td>Sign In</td>
<td>Information Technology</td>
<td>Details</td>
</tr>
<tr>
<td>Air New Zealand</td>
<td>Sign In</td>
<td>Travel &amp; Tourism</td>
<td>Details</td>
</tr>
<tr>
<td>Albert Heijn</td>
<td>Sign In</td>
<td>eCommerce</td>
<td>Details</td>
</tr>
<tr>
<td>Amazon</td>
<td>Sign In</td>
<td>eCommerce</td>
<td>Details</td>
</tr>
<tr>
<td>Apple</td>
<td>Sign In</td>
<td>Information Technology</td>
<td>Details</td>
</tr>
<tr>
<td>Arcave</td>
<td>Sign In</td>
<td>Social</td>
<td>Details</td>
</tr>
<tr>
<td>Arpeli</td>
<td>Sign In</td>
<td>Finance</td>
<td>Details</td>
</tr>
<tr>
<td>au</td>
<td>Sign In</td>
<td>Information Technology</td>
<td>Details</td>
</tr>
</tbody>
</table>
Multi-factor vs. Multi-step

**Factor:** something you know/have/are

**Step:** user-specific action

Type password, tap fingerprint reader, press security key, look at camera, …

**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**

Phone Face ID ➔ **two factors, one step**

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 (shared secret key or private key)
- Can use symmetric (e.g., Kerberos) or public key (e.g., U2F, passkeys) 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
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.
This is a normal message in a normal conversation.

Now Alice is going to reinstall.

As soon as Alice reinstalled, I saw the notice above. Impressive.

Now Alice has uninstalled, and this message is being transmitted before Alice reinstalls.

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

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.
Certificates

How can we distribute “trusted” public keys?
   Public directory → risk of forgery and tampering, scalability issues
   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, manage, 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 a browser warning if it has been certified by a trusted CA

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

Operating systems and browsers are pre-configured with tens/hundreds of trusted root certificates (more on this in the TLS lecture)

Single point of failure: CAs can be compromised!
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.
Web of Trust (mainly used in PGP for encrypted email – future lecture)

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!)

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: challenging to create a WoT for the whole world
WoT: Finding Public Keys

Public PGP key servers

- pgp.mit.edu
- keyservers.pgp.com

Cache certificates from received emails

Integration with user management systems (LDAP, IAM>IDP)

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
Online Social “Tracking”
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, …

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

Avoid SMS if possible! Use an authenticator app or even better U2F (or passkeys)

Remove phone number from account after authenticator/U2F setup

Store your backup codes/backup key 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