

2550 Intro to cybersecurity

L3

abhi shelat

What does it mean to attack a *system*?



abhi



Enter Password

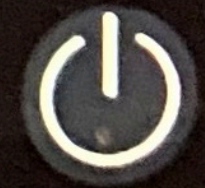
Touch ID or Enter Password



Sleep



Restart



Shut Down



LTE



Touch ID or Enter Passcode



1

2

3

ABC

DEF

4

5

6

GHI

JKL

MNO

7

8

9

PQRS

TUV

WXYZ

0

Emergency

Cancel



Northeastern University Information Technology Services

Welcome to NUwave-guest

Log in to Northeastern's unsecured wireless network NUwave-guest using the username and password you received via text message.

Need to register? [Click here.](#)

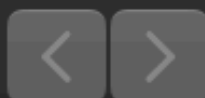
One Day Conference Login [Click here.](#)

Have a myNEU login? You must log into NUwave - the secure wireless network.

NUwave-guest Login

Username:

Password:





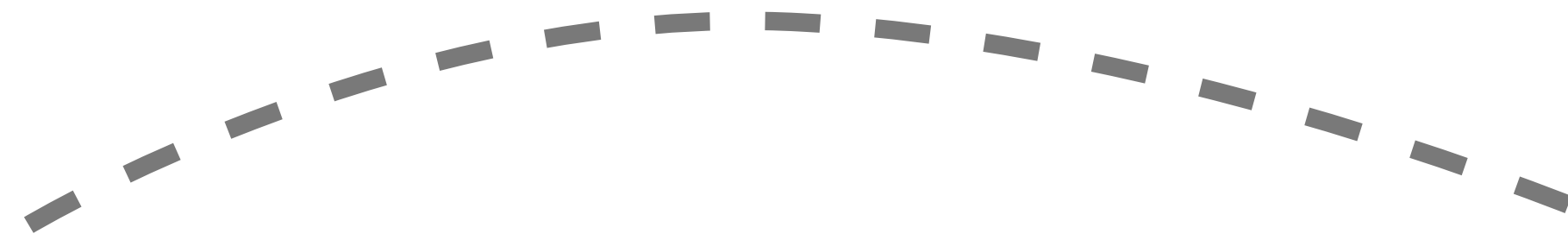
Authentication

- **Authentication** is the process of verifying an actor's **identity**
- **Critical for security of systems**
 - Permissions, capabilities, and access control are all contingent upon knowing the identity of the actor
- Typically parameterized as a **username** and a **secret**
 - The secret attempts to limit unauthorized access
- Desirable properties of secrets include being *unforgeable*, *unguessable*, and *revocable*

Passwords

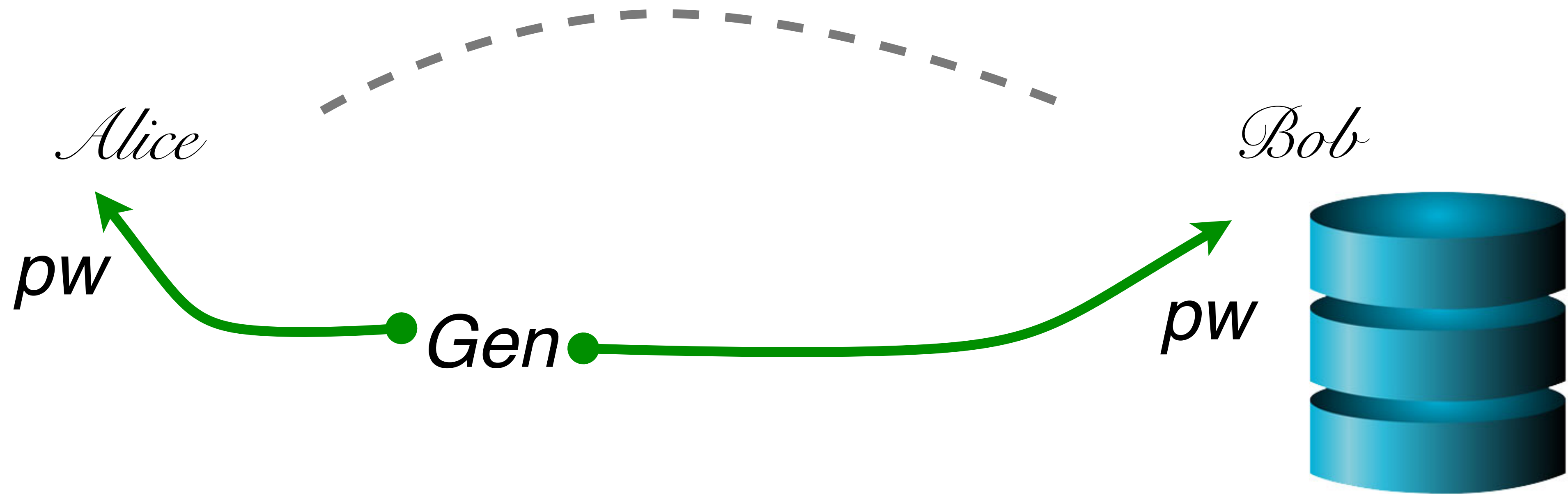
Main problem:

Alice



Bob

Passwords





Create your Google Account

First name Last name

Username @gmail.com

You can use letters, numbers & periods

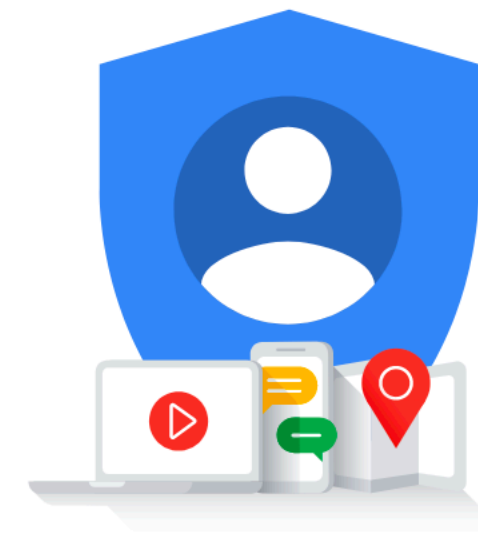
[Use my current email address instead](#)

Password Confirm

Use 8 or more characters with a mix of letters, numbers & symbols

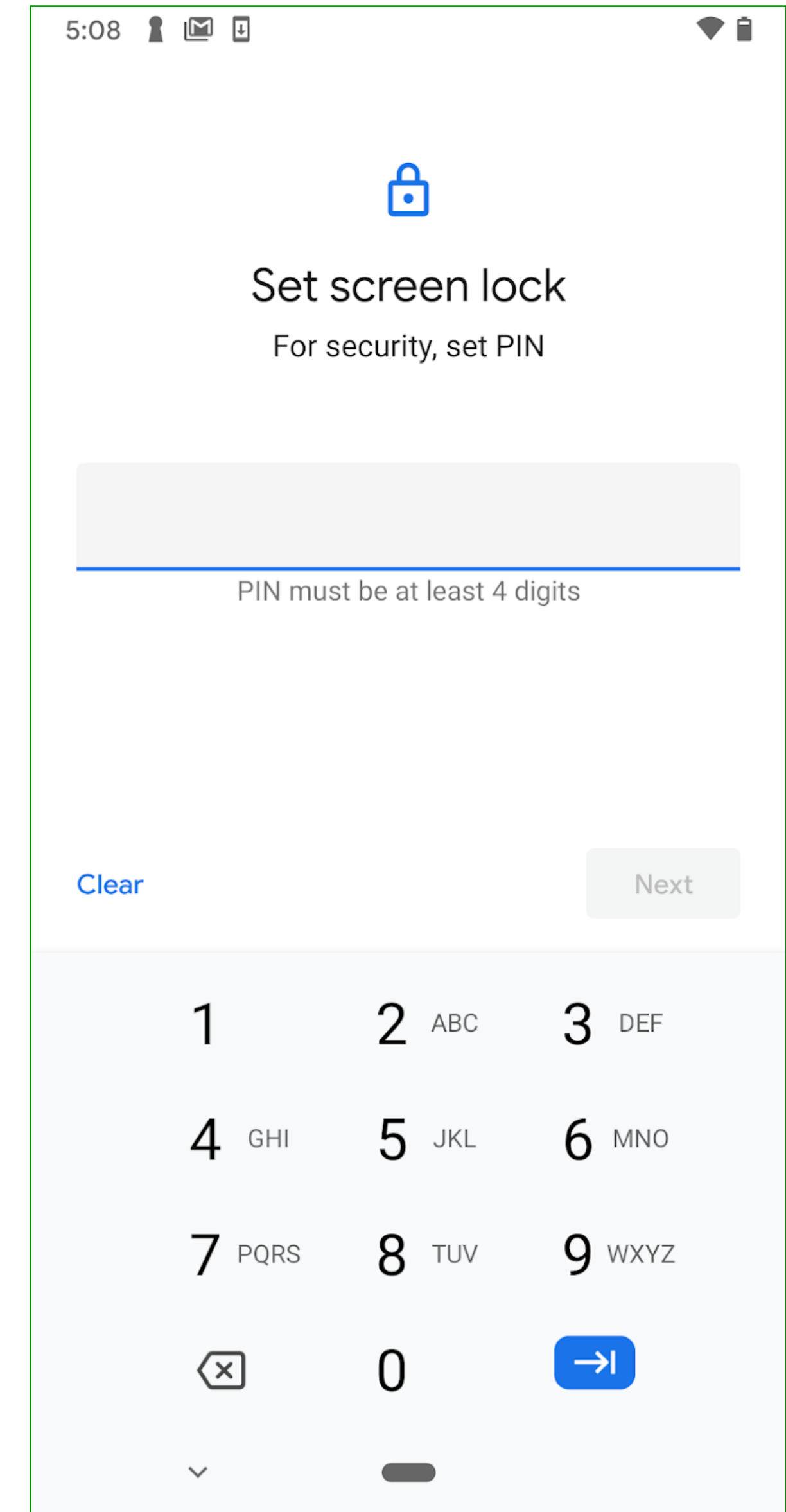
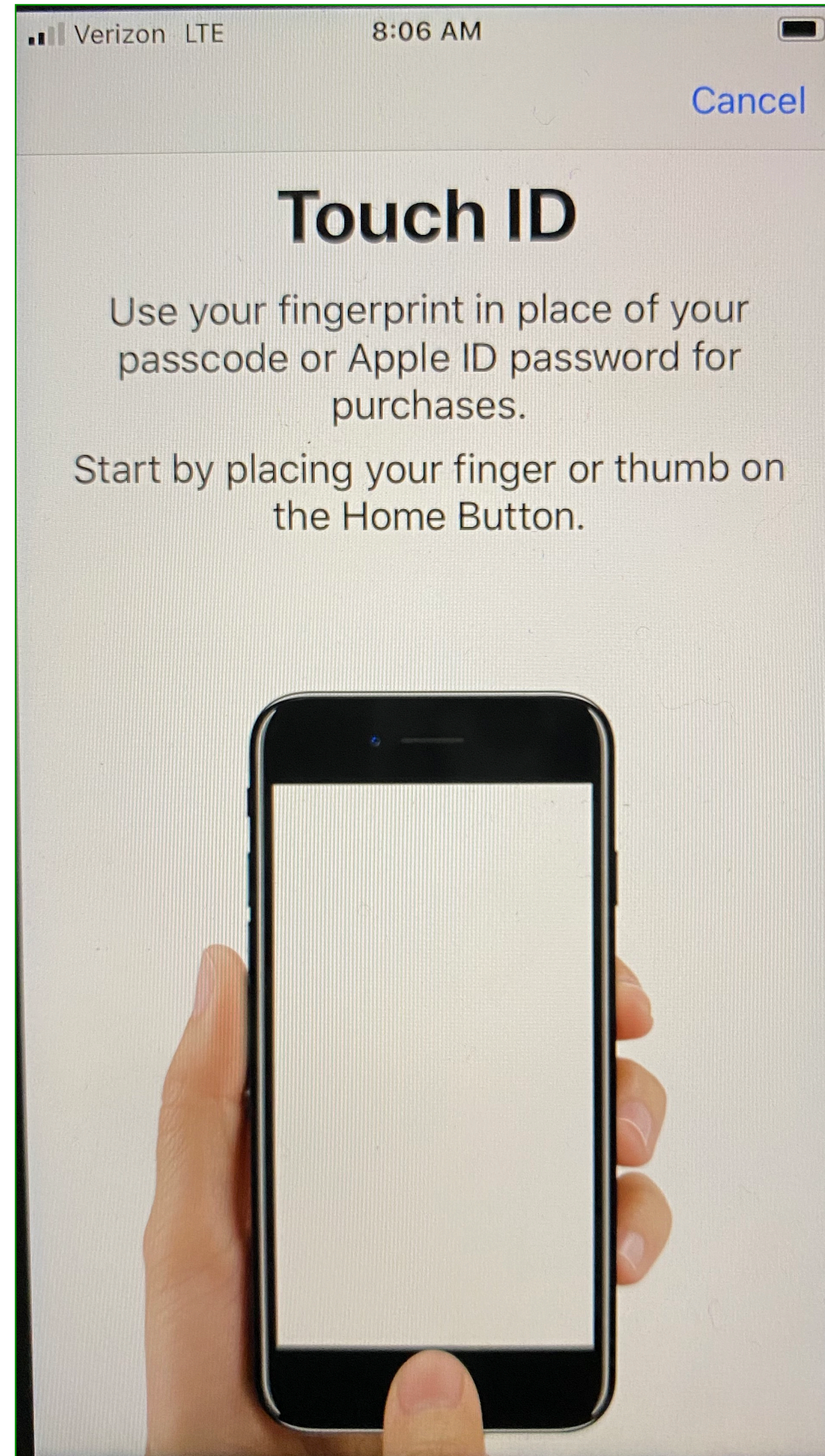
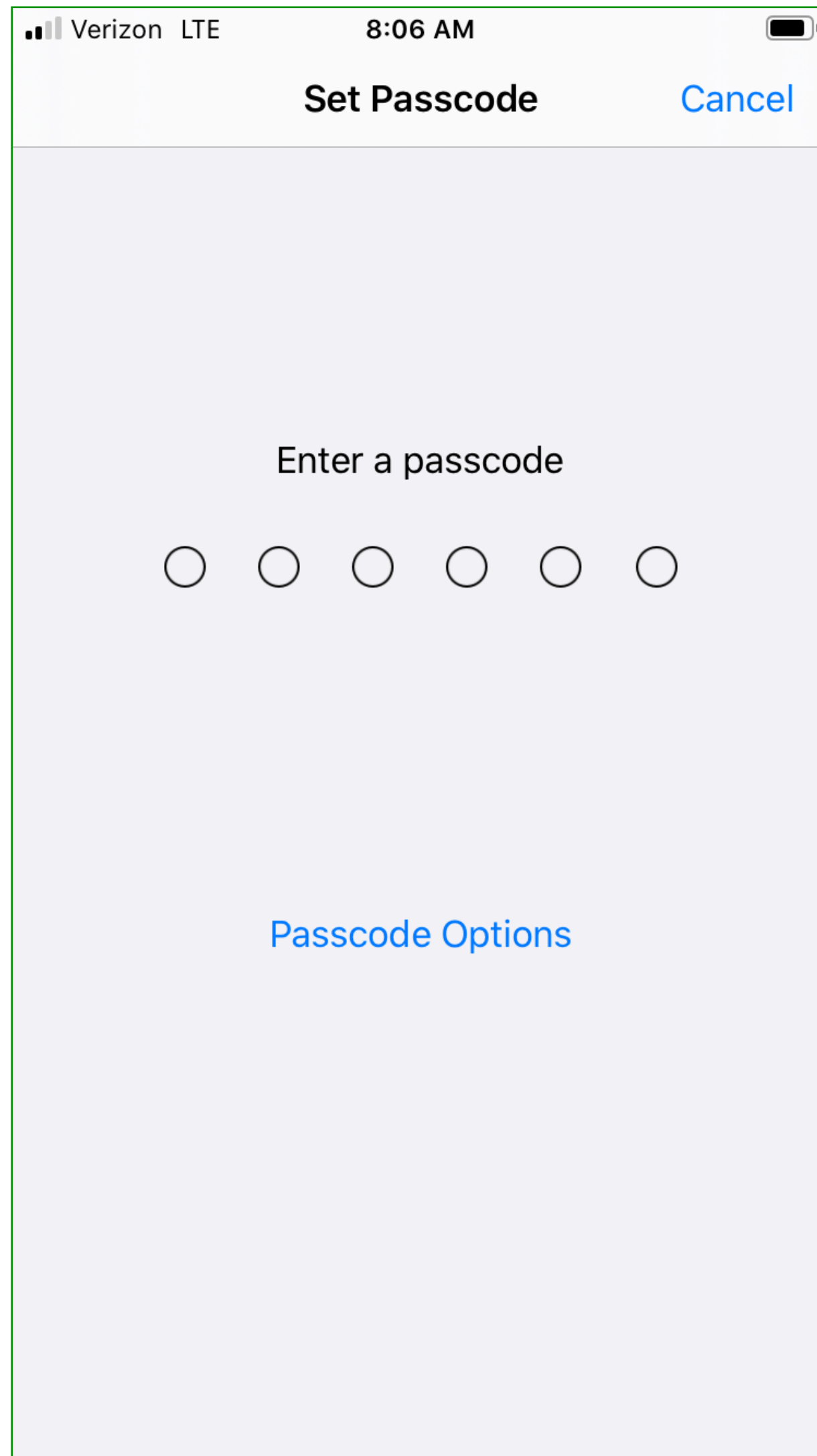
[Sign in instead](#)

[Next](#)



One account. All of Google working for you.

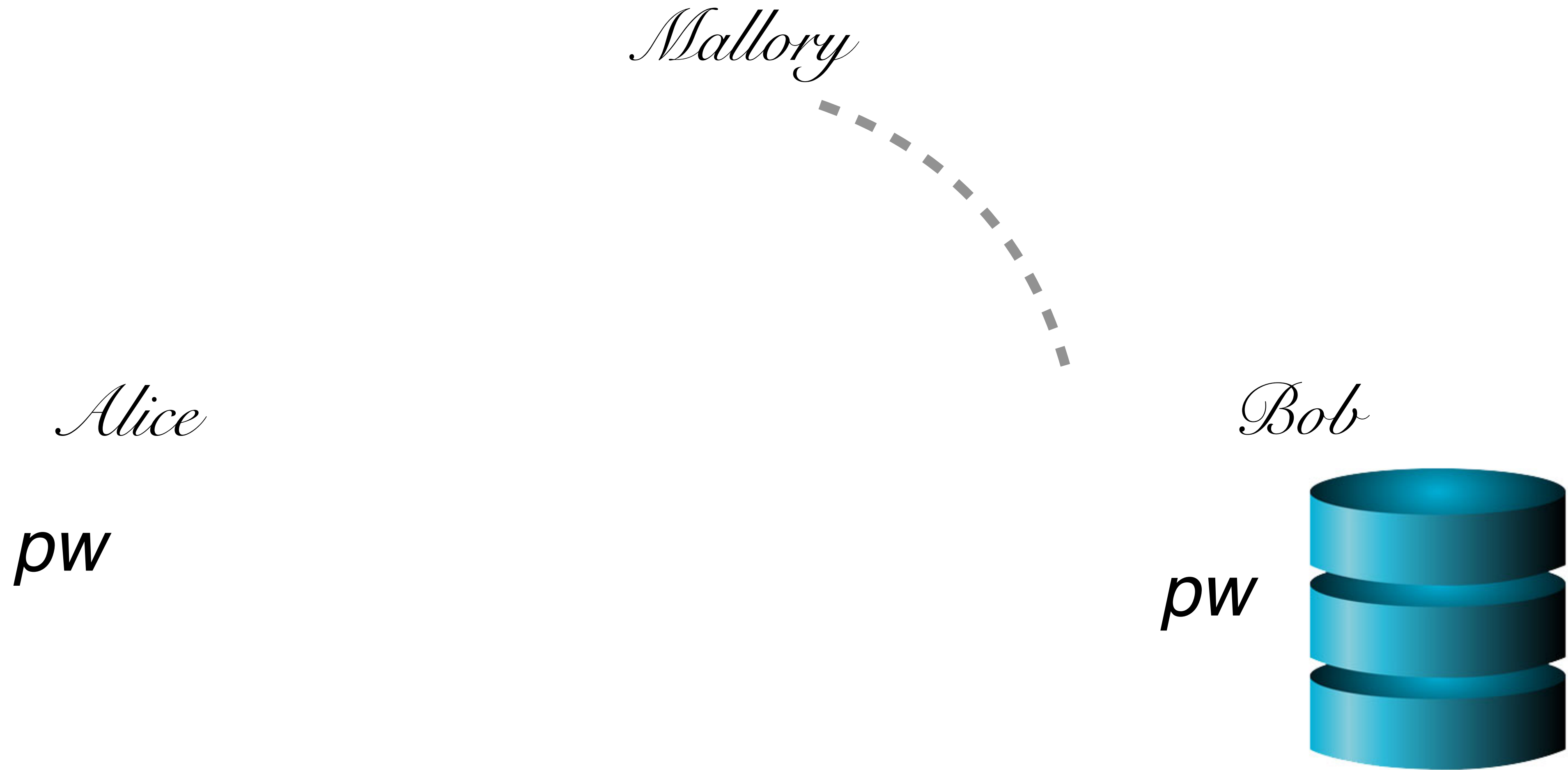
PIN setup



Passwords: Alice always succeeds



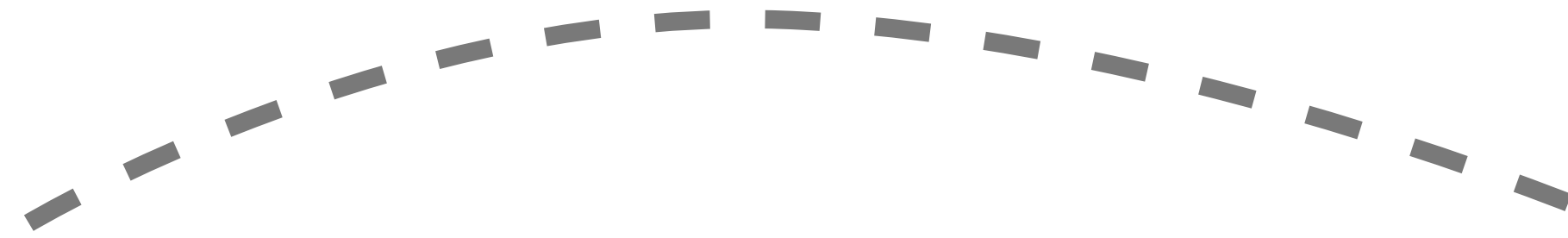
Passwords: Others do not succeed



Natural authenticators

First problem: How does Bob *check* Alice's password?

Alice



Bob

pw

pw



Checking Passwords

- System must validate passwords provided by users
- Thus, passwords must be *stored* somewhere
- Basic storage: plain text

password.txt	
Alice	p4ssw0rd
Eve	i heart doggies
Charlie	93Gd9#jv*0x3N
bob	security

Operating
Systems

R. Stockton Gaines
Editor

Password Security: A Case History

Robert Morris and Ken Thompson
Bell Laboratories

This paper describes the history of the design of the password security scheme on a remotely accessed time-sharing system. The present design was the result of countering observed attempts to penetrate the system. The result is a compromise between extreme security and ease of use.

Key Words and Phrases: operating systems, passwords, computer security

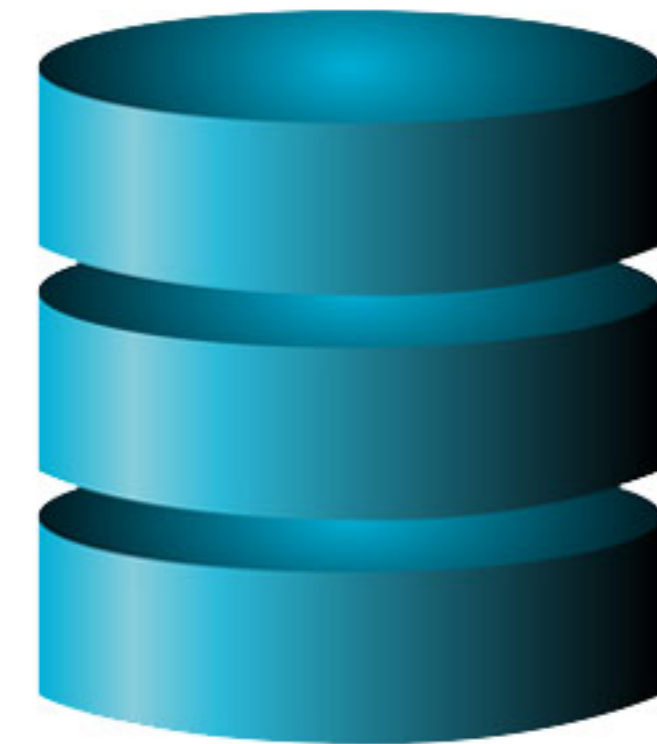
CR Categories: 2.41, 4.35

"The UNIX system was first implemented with a password file that contained the actual passwords of all the users, and for that reason the password file had to be heavily protected against being either read or written. Although historically, this had been the technique used for remote-access systems, it was completely unsatisfactory for several reasons."

Attacks against the Password Model

Mallory

Bob



{username: pwd}

password.txt

Alice	p4ssw0rd
Eve	i heart doggies
Charlie	93Gd9#jv*0x3N
bob	security

Problem: Password File Theft

- Attackers often compromise systems
- They may be able to steal the password file
 - Linux: /etc/shadow
 - Windows: c:\windows\system32\config\sam
- If the passwords are plain text, what happens?
 - The attacker can now log-in as any user, including root/administrator
- **Passwords should never be stored in plain text**

RockYou Hack: From Bad To Worse

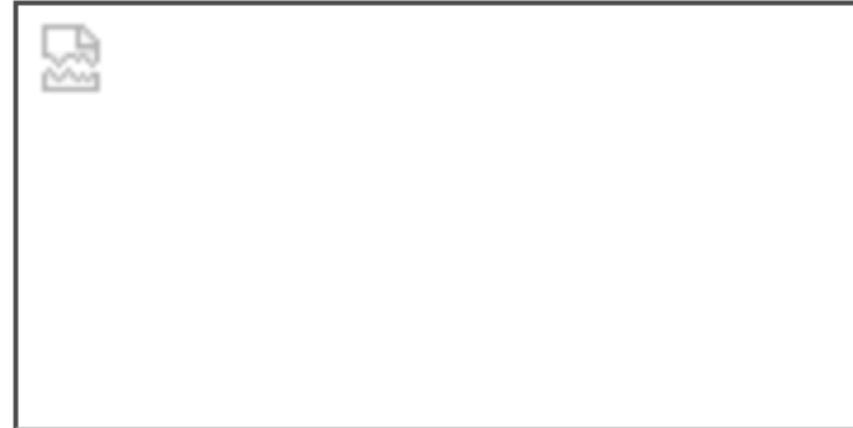


Nik Cubrilovic




@nikcub / 2:42 am EST • December 15, 2009

Comment



[Earlier today news spread](#) that social application site [RockYou](#) had suffered a data breach that

resulted in the exposure of over 32 Million user accounts. To compound the severity of the security breach, it was found that **RockYou**  are storing all user account data in plain text in their database, exposing all that information to attackers. RockYou have yet to inform users of the breach, and their blog is eerily silent – but the details of the security breach are going from bad to worse.



Data UserAccount [32603388]

=====

- 1|jennaplanerunner@hotmail.com|mek****|myspace|0|bebo.com
- 2|phdlance@gmail.com|mek****|myspace|1|
- 3|jennaplanerunner@gmail.com|mek****|myspace|0|
- 5|teamsmackage@gmail.com|pro****|myspace|1|
- 6|ayul@email.com|kha****|myspace|1|tagged.com
- 7|guera_n_negro@yahoo.com|emi****|myspace|0|
- 8|beyootifulgirl@aol.com|hol****|myspace|1|
- 9|keh2oo8@yahoo.com|cai****|myspace|1|
- 10|mawabiru@yahoo.com|pur****|myspace|1|
- 11|jodygold@gmail.com|att****|myspace|1|
- 12|aryan_dedboy@yahoo.com|iri****|myspace|0|
- 13|moe_joe_25@yahoo.com|725****|myspace|1|
- 14|xxxnothingbutme@aol.com|1th****|myspace|0|
- 15|meandcj069@yahoo.com|too****|myspace|0|
- 16|stacey_chim@hotmail.com|cxn****|myspace|1|
- 17|barne1en@cmich.edu|ilo****|myspace|1|
- 18|reo154@hotmail.com|ecu****|myspace|1|
- 19|natapappaslie@yahoo.com|tor****|myspace|0|
- 20|ypiogirl@aol.com|tob****|myspace|1|
- 21|brittanyleigh864@hotmail.com|bet****|myspace|1|myspace.com
- 22|topenga68@aol.com|che****|myspace|0|
- 23|marie603412@yahoo.com|cat****|myspace|0|
- 24|mellowchick41@aol.com|chu****|myspace|0|

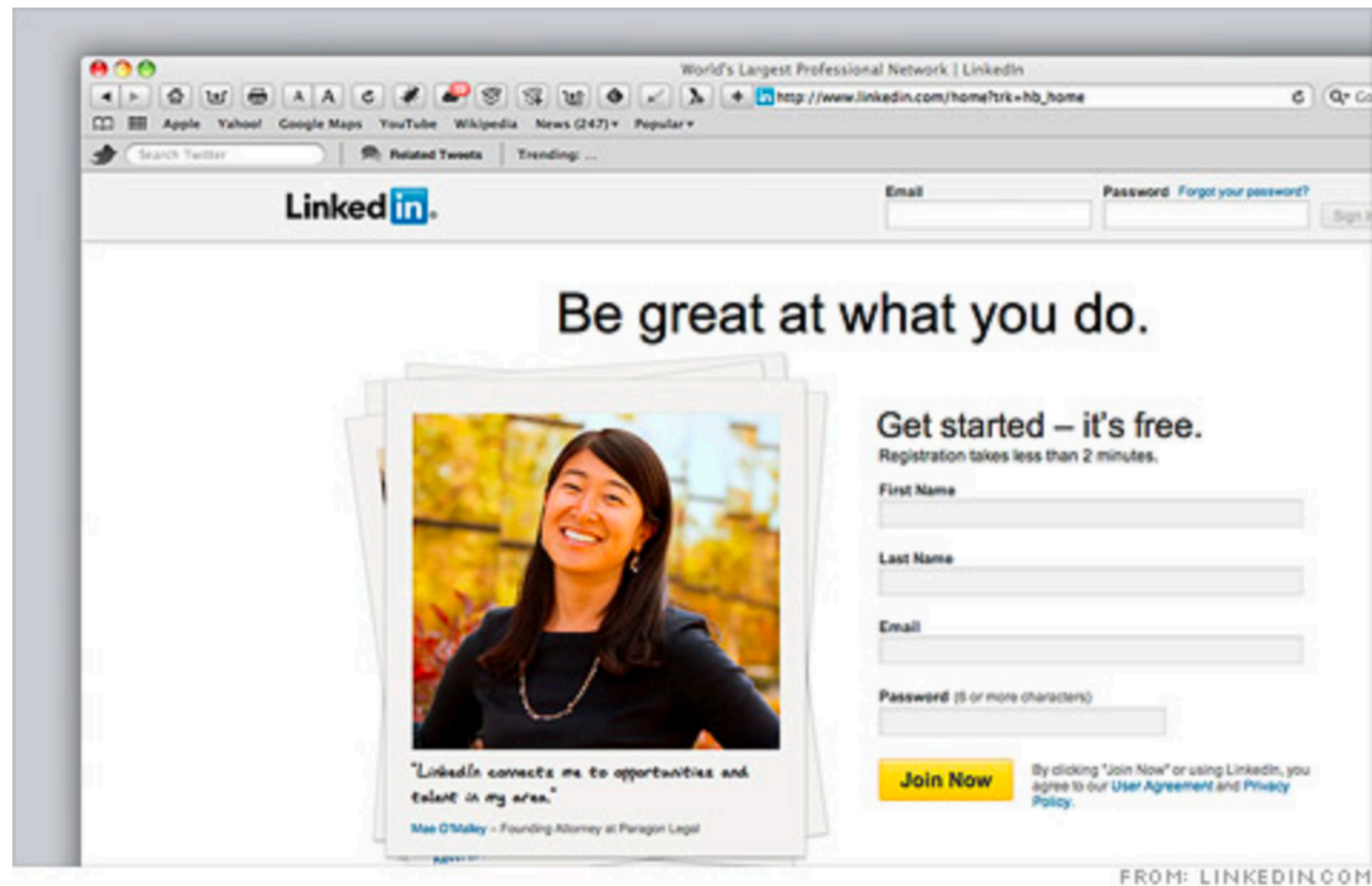
Pwd breaches

CNN Money

THE CYBERCRIME ECONOMY

More than 6 million LinkedIn passwords stolen

By David Goldman @CNMMoneyTech June 7, 2012: 9:34 AM ET



Researchers say a stash of what appear to be LinkedIn passwords were protected by a weak security scheme.

NEW YORK (CNMMoney) -- Russian hackers released a giant list of passwords this week, and on Wednesday security researchers identified their likely source: business social networking site LinkedIn.

Password Security: A Case History

Robert Morris and Ken Thompson
Bell Laboratories

This paper describes the history of the design of the password security scheme on a remotely accessed time-sharing system. The present design was the result of countering observed attempts to penetrate the system. The result is a compromise between extreme security and ease of use.

Key Words and Phrases: operating systems, passwords, computer security

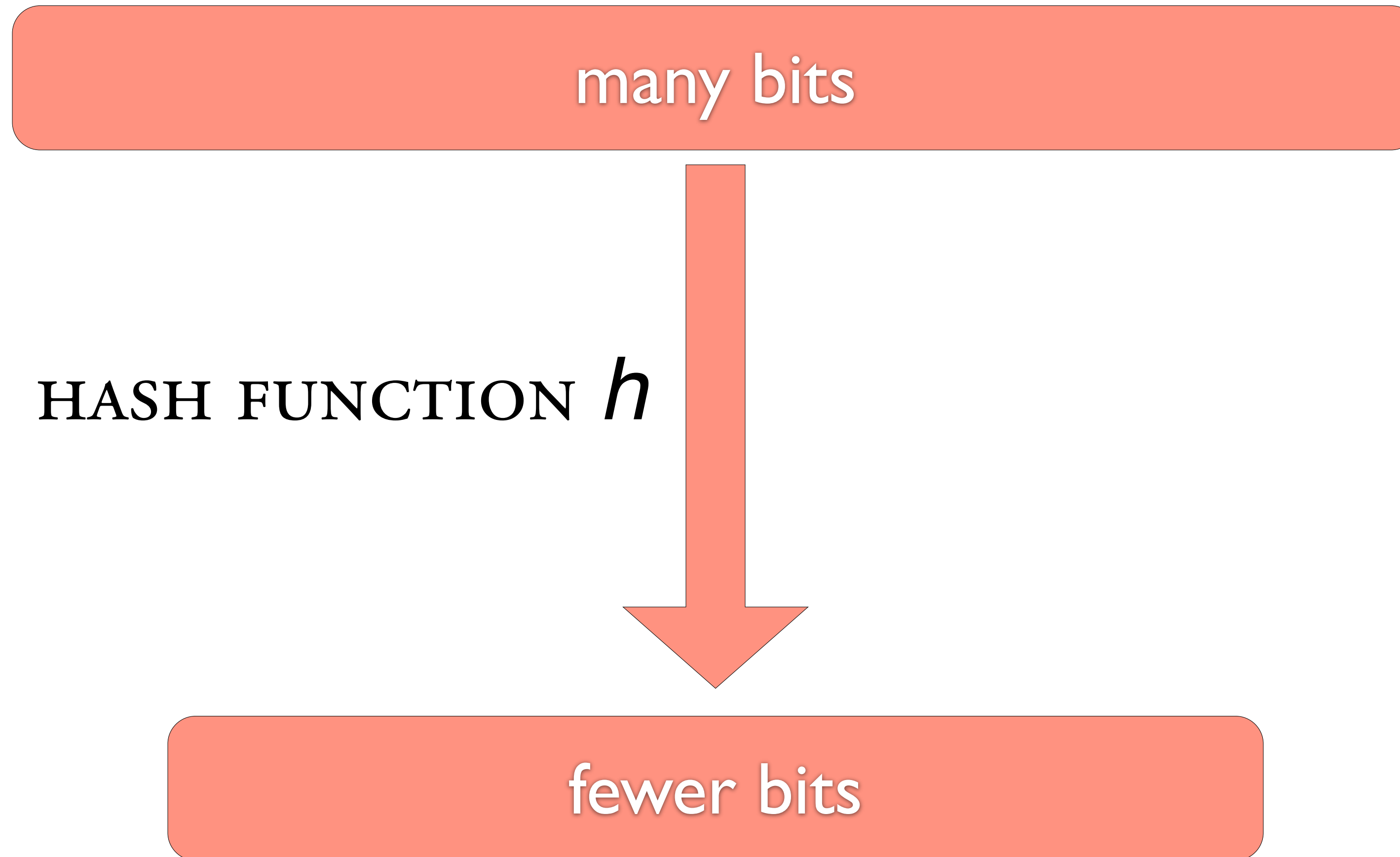
CR Categories: 2.41, 4.35

“The obvious solution is to arrange that the passwords not appear in the system at all, and it is not difficult to decide that this can be done by encrypting each user's password, putting only the encrypted form in the password file, and throwing away his original password (the one that he typed in). When the user later tries to log in to the system, the password that he types is encrypted and compared with the encrypted version in the password file. If the two match, his login attempt is accepted.”

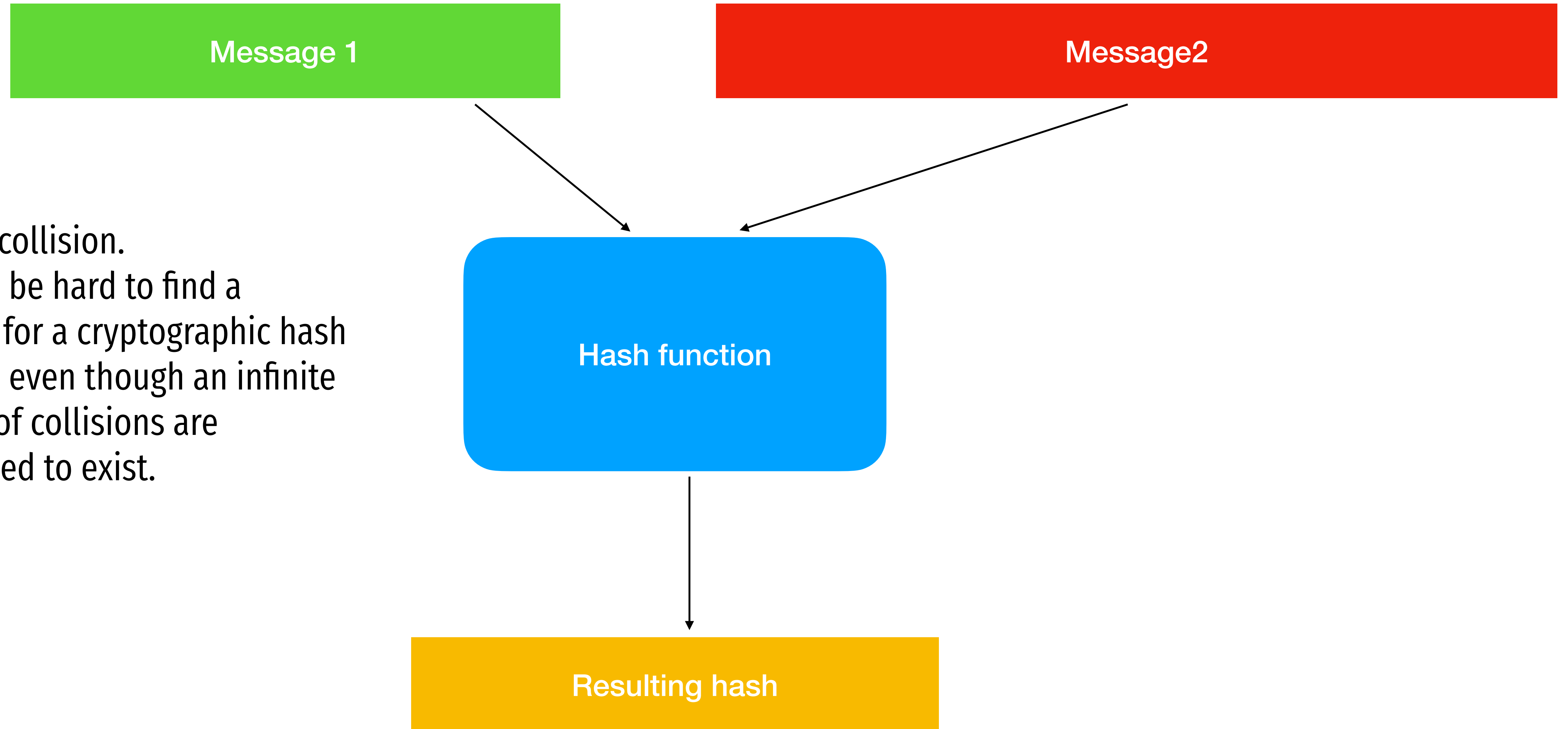
Hashed Passwords

- **Key idea: store “hashed” versions of passwords**
 - Use one-way cryptographic hash functions
 - Examples: MD5, SHA1, SHA256, SHA512, bcrypt, PBKDF2, scrypt

Goal of a hash function



Goal of a hash function: Collision resistance



This is a collision.
It should be hard to find a collision for a cryptographic hash function, even though an infinite number of collisions are guaranteed to exist.

MD5 is a broken hash function

```
$ md5 -s security  
MD5 ("security") = e91e6348157868de9dd8b25c81aebfb9
```

```
$ md5 -s Security  
MD5 ("Security") = 2fae32629d4ef4fc6341f1751b405e45
```

```
$ md5 -s Security1  
MD5 ("Security1") = 8d01bda744a7a6392d3393e0ece561e8
```


```
$ echo -n "security" | shasum  
8eec7bc461808e0b8a28783d0bec1a3a22eb0821 -
```

```
$ echo -n "security" | shasum -a 256  
5d2d3ceb7abe552344276d47d36a8175b7aeb250a9bf0bf00e850cd23ecf2e43 -
```

Hashed Passwords

- **Key idea: store “hashed” versions of passwords**
 - Use one-way cryptographic hash functions
 - Examples: MD5, SHA1, SHA256, SHA512, bcrypt, PBKDF2, scrypt
- **Cryptographic hash function transform input data into scrambled output data**
 - Deterministic: $\text{hash}(A) = \text{hash}(A)$
 - Collision resistant
 - Locating A' such that $\text{hash}(A) = \text{hash}(A')$ takes a long time (hopefully)
 - Example: 2^{21} tries for md5

Hashed Password Example


User: Charlie



MD5('p4ssw0rd') =
2a9d119df47ff993b662a8ef36f9ea20



MD5('2a9d119df47ff993b662a8ef36f9ea20')
= b35596ed3f0d5134739292faa04f7ca3



hashed_password.txt	
charlie	2a9d119df47ff993b662a8ef36f9ea20
greta	23eb06699da16a3ee5003e5f4636e79f
alice	98bd0ebb3c3ec3fbe21269a8d840127c
bob	e91e6348157868de9dd8b25c81aebfb9

Attacking Password Hashes

- Recall: cryptographic hashes are collision resistant
 - Locating A' such that $\text{hash}(A) = \text{hash}(A')$ takes a long time (hopefully)
- Are hashed password secure from cracking?
 - **No!**
- Problem: users choose poor passwords
 - Most common passwords: 123456, password
 - Username: cbw, Password: cbw
- Weak passwords enable **dictionary attacks**

The authors have conducted experiments to try to determine typical users' habits in the choice of passwords when no constraint is put on their choice. The results were disappointing, except to the bad guy. In a collection of 3,289 passwords gathered from many users over a long period of time,

15 were a single ASCII character;

72 were strings of two ASCII characters;

464 were strings of three ASCII characters;

477 were strings of four alphanumerics;

706 were five letters, all upper-case or all lower-case;

605 were six letters, all lower-case.

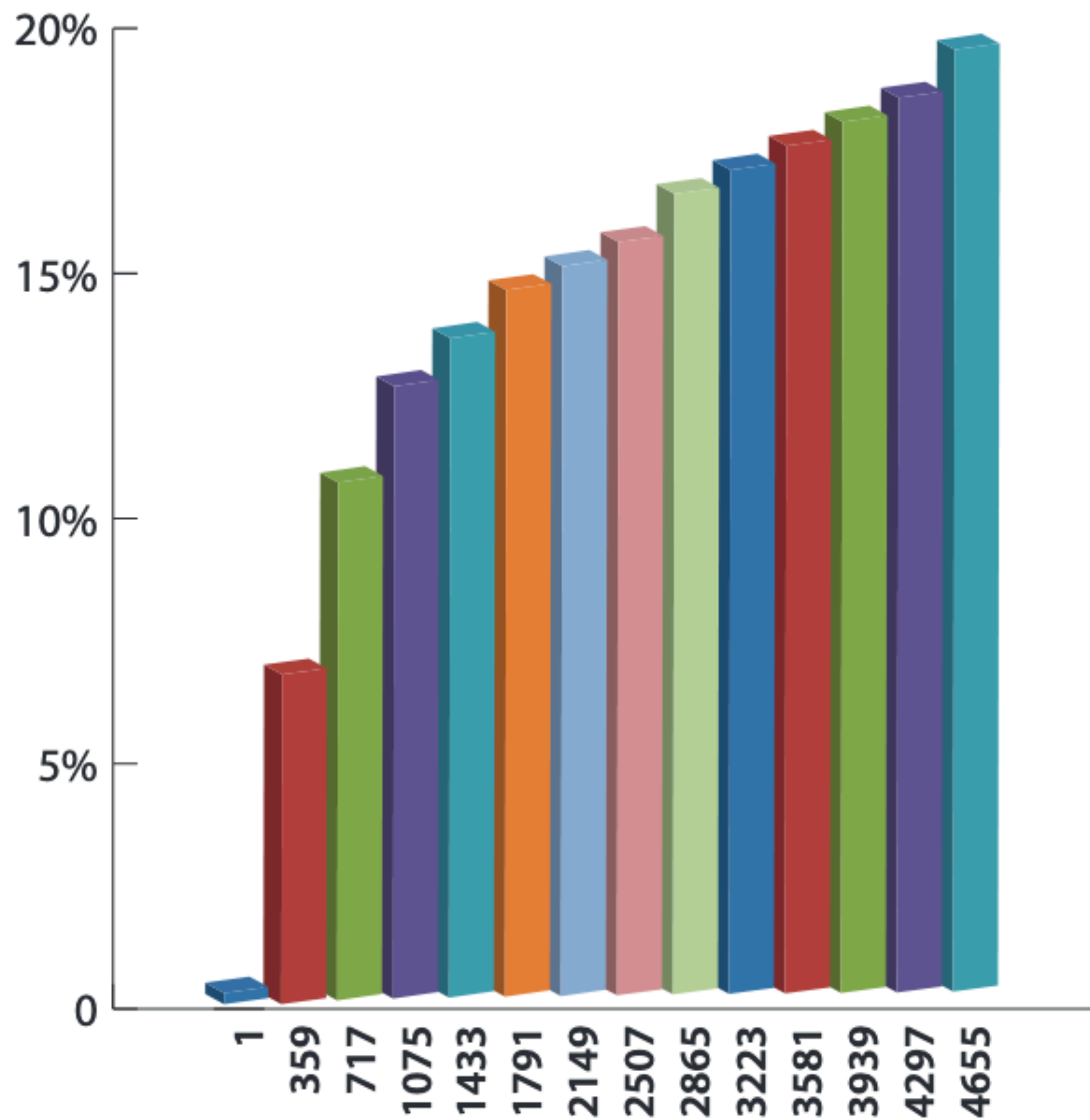
An additional 492 passwords appeared in various available dictionaries, name lists, and the like. A total of 2,831 or 86 percent of this sample of passwords fell into one of these classes.

From Rockyou breach

Rank	Password	Number of Users with Password (Absolute)
1	123456	290731
2	12345	79078
3	123456789	76790
4	Password	61958
5	iloveyou	51622
6	princess	35231
7	rockyou	22588
8	1234567	21726
9	12345678	20553
10	abc123	17542

Password Popularity—Top 20

Rank	Password	Number of Users with Password (Absolute)
11	Nicole	17168
12	Daniel	16409
13	babygirl	16094
14	monkey	15294
15	Jessica	15162
16	Lovely	14950
17	michael	14898
18	Ashley	14329
19	654321	13984
20	Qwerty	13856



Accumulated Percent of Dictionary Attack Success

Most Common Passwords

Rank	2013	2014
1	123456	123456
2	password	password
3	12345678	12345
4	qwerty	12345678
5	abc123	qwerty
6	123456789	123456789
7	111111	1234
8	1234567	baseball
9	iloveyou	dragon
10	adobe123	football

2012: 6.5 million hashes leaked onto Internet 90% cracked in 2 weeks

2016: 177.5 million more hashes leaked 98% cracked in 1 week

2012 LinkedIn Breach had 117 Million Emails and Passwords Stolen, Not 6.5M


May 18, 2016



Long time users of LinkedIn users may very well need to change their passwords once more



Related Posts

- Web Skimming Attack on Blue
-  Bear Affects School Admin

by Paul Ducklin



One month ago today, we wrote about Adobe's [giant data breach](#).

As far as anyone knew, including Adobe, it affected about 3,000,000 customer records, which made it sound pretty bad right from the start.



But worse was to come, as recent updates to the story bumped the number of affected customers to a [whopping 38,000,000](#).

We took Adobe to task for a lack of clarity in its breach notification.

OUR COMPLAINT

One of our complaints was that Adobe said that it had lost *encrypted* passwords, when we thought the company ought to have said that it had lost

```

4464 ① User ID yahoo.com-|-g2B6PhWEH36 ⑤ Password hint try: qwerty123 --
4465-|-|-xxxxx@jcom.home.ne.jp-|-Eh5tLomK+N+82csoVwU9bw==|-|?????|--
4466-|-|-xx@hotmail.com-|-ahw2b2BELzgrTWYvQGn+kw==|-quiero a...|--
4467-|-|-xxx@yahoo.com-|-leMTcMPEPcjioxG6CatHBw==|-|--
4468-|-username ② Username e.com-|-2GthVrmsERzioxG6CatHBw==|-|--
4469-|-|-xxxxx@yahoo.com-|-4LSlo772tH4= ④ Password data (base64)
4470-|-|-xxx@hotmail.com-|-xxxxx@xxxxx.com-|-xG6CatHBw==|-|--
4471-|-|-xxxx@yahoo.com ③ Email address xG6CatHBw==|-myspace|--
4471-|-|-xxx@hotmail.com-|-kby1918wDrrioxG6CatHBw==|-regular|--

```

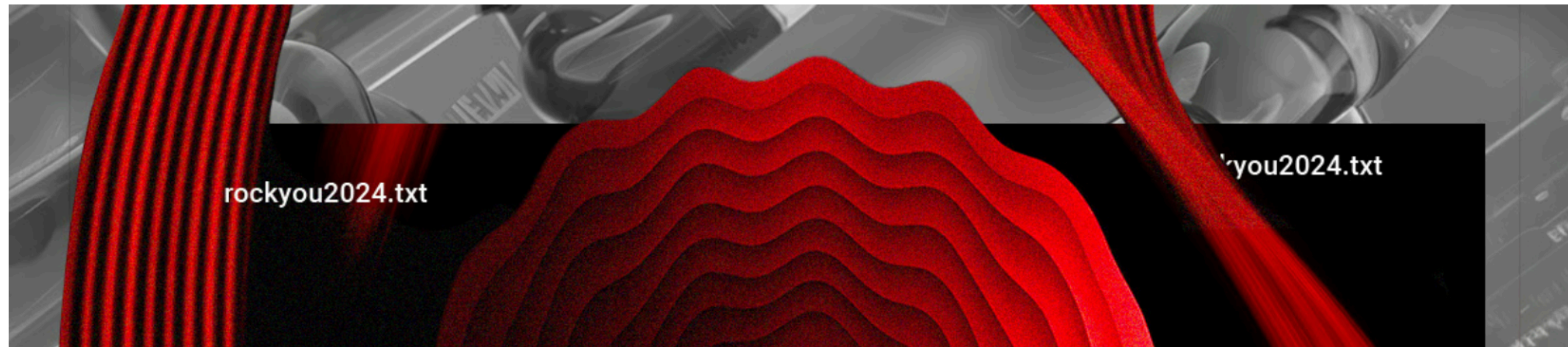
Adobe password data	Password hint	
110edf2294fb8bf4	-> numbers 123456	
110edf2294fb8bf4	-> ==123456	① 123456
110edf2294fb8bf4	-> c'est "123456"	
8fda7e1f0b56593f e2a311ba09ab4707	-> numbers	
8fda7e1f0b56593f e2a311ba09ab4707	-> 1-8	② 12345678
8fda7e1f0b56593f e2a311ba09ab4707	-> 8digit	
2fca9b003de39778 e2a311ba09ab4707	-> the password is password	
2fca9b003de39778 e2a311ba09ab4707	-> password	③ password
2fca9b003de39778 e2a311ba09ab4707	-> rhymes with assword	
e5d8efed9088db0b	-> q w e r t y	
e5d8efed9088db0b	-> ytrewq tagurpidi	④ qwerty
e5d8efed9088db0b	-> 6 long qwert	
ecba98cca55eabc2	-> sixxone	
ecba98cca55eabc2	-> 1*6	⑤ 111111
ecba98cca55eabc2	-> sixones	

RockYou2024: 10 billion passwords leaked in the largest compilation of all time

Updated on: July 04, 2024 12:33 PM  4

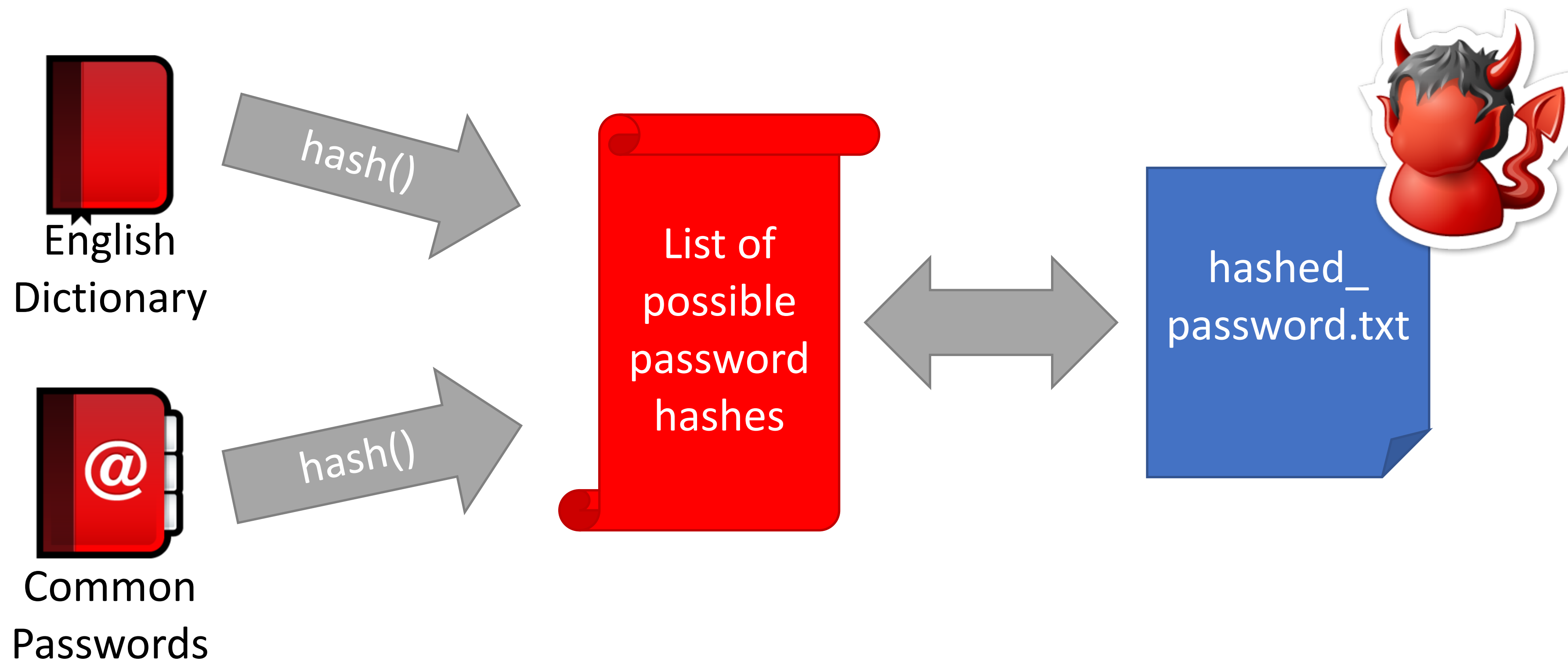


Vilius Petkauskas, Deputy Editor



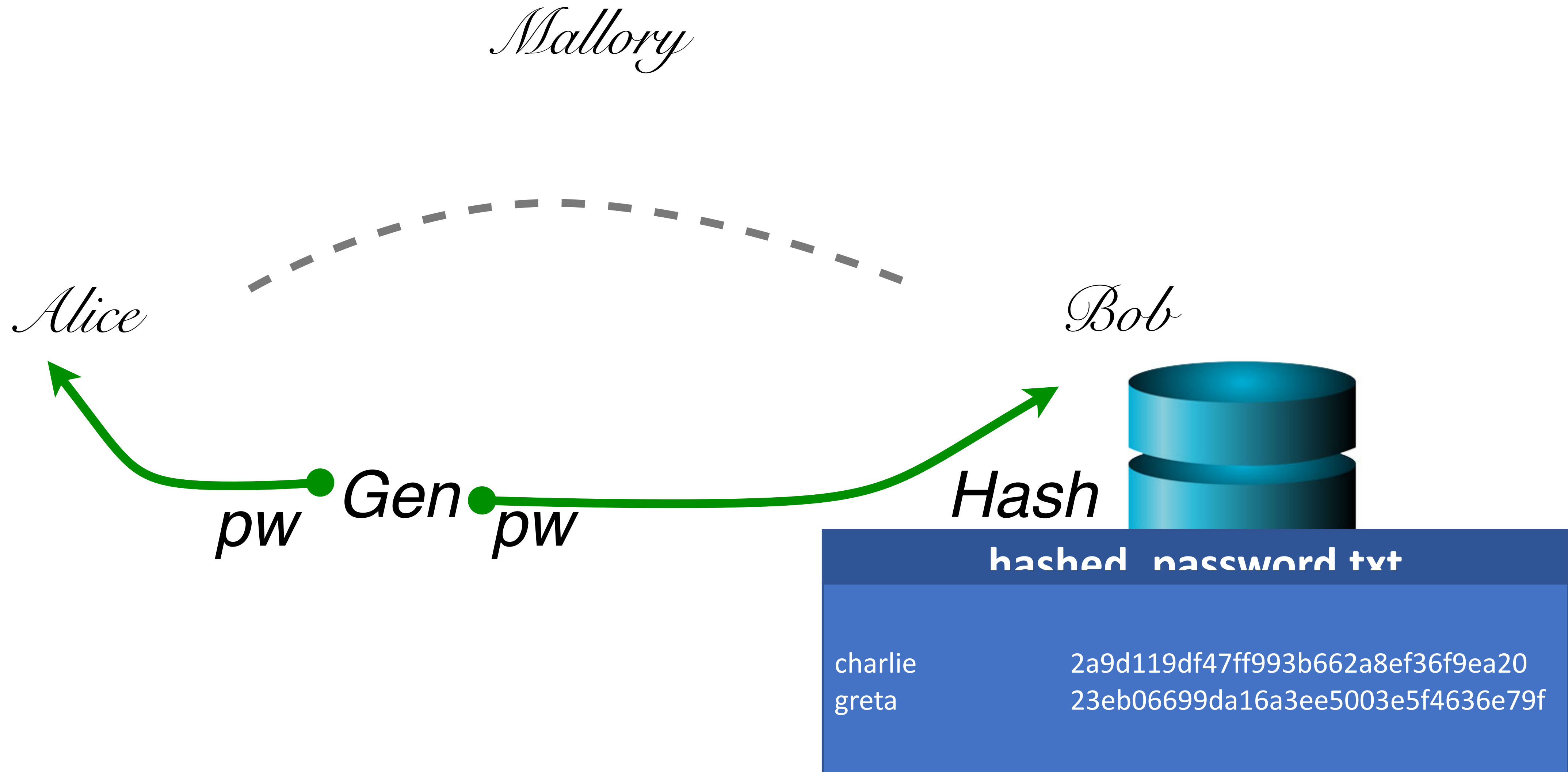
The king is dead. Long live the king. Cybernews researchers discovered what appears to be the largest password compilation with a staggering 9,948,575,739 unique plaintext passwords. The file with the data, titled **rockyou2024.txt**, was posted on July 4th by forum user ObamaCare.

Dictionary Attacks

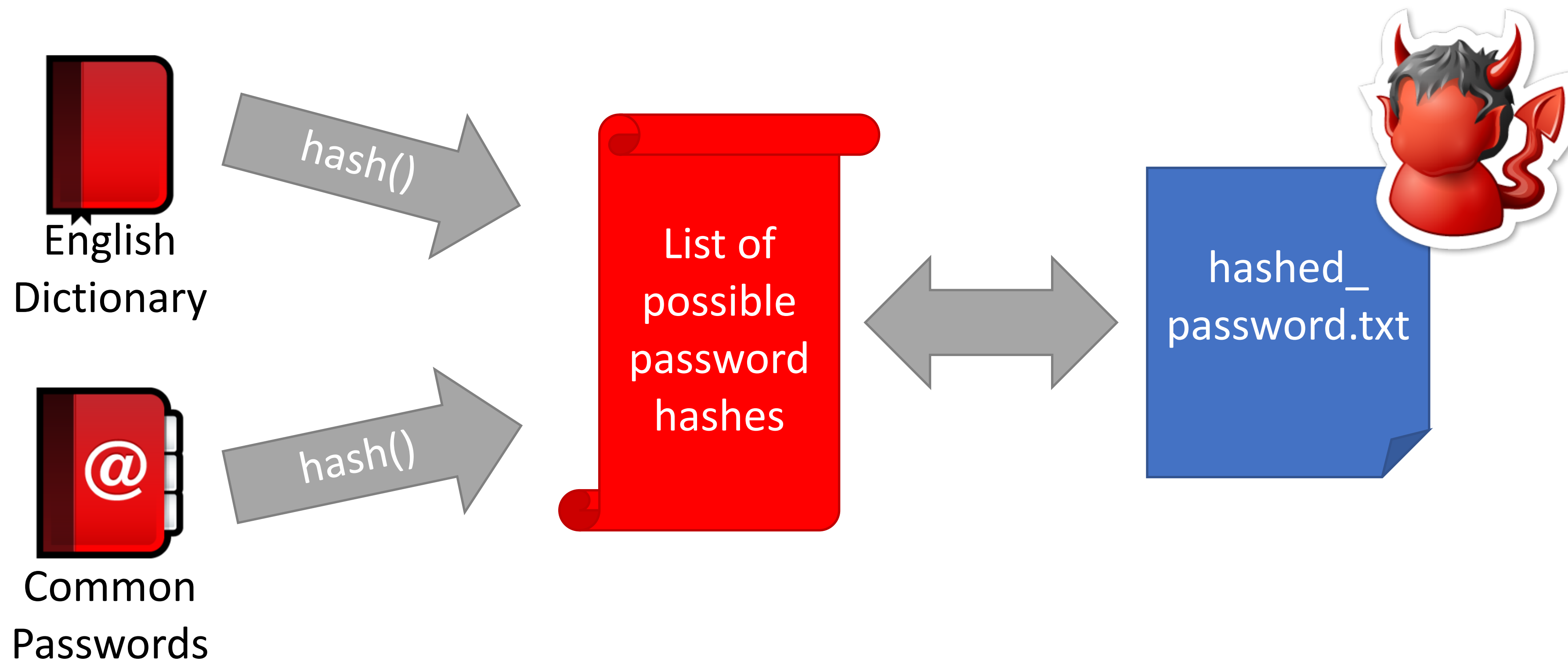


- Common for 60-70% of hashed passwords to be cracked in <24 hours

Attack 1



Dictionary Attacks



- Common for 60-70% of hashed passwords to be cracked in <24 hours

Brute force attack estimates

How big is the alphabet from which pwd are chosen?

Brute force attack estimates

How big is the alphabet from which pwd are chosen?

95 symbols

How long is a password?

Size of password domain:

Brute force attack estimates

Size of password domain: 95^8 6,634,204,312,890,625

<https://diskprices.com/?locale=us&condition=new>

Price per TB	Price	Capacity	Warranty	Form Factor	Technology	Condition	Affiliate Link
\$8.124	\$130	16 TB	5 years	Internal	SAS	New	MDD 16TB 7200RPM 256MB Cache SAS 12.0Gb/s 3.5inch Internal Enterprise Hard Drive (MDD16TSAS25672E) - [NOT a SATA HDD]
\$9.166	\$110	12 TB	3 years	Internal 3.5"	HDD	New	MDD 12TB 7200RPM SATA 6Gb/s 256MB Cache 3.5inch Internal Desktop Hard Drive, MD12TBGSA25672, Mechanical Hard Disk
\$10.00	\$100	10 TB	3 months	Internal	SAS	New	HUH721010AL4200 HGST Ultrastar He10 10TB 7200RPM SAS 12Gbps 256MB 2018
\$10.48	\$84	8 TB	3 years	Internal 3.5"	HDD	New	MaxDigitalData 8TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5inch Internal Hard Drive for Surveillance (MD8000GSA25672DVR)
\$10.75	\$129	12 TB	3 years	Internal 3.5"	HDD	New	Seagate 12TB IronWolf NAS SATA Hard Drive 6Gb/s 256MB Cache 3.5-Inch Internal Hard Drive for NAS Servers, Personal Cloud Storage (ST12000VN0007)
\$10.81	\$86	8 TB	3 years	Internal 3.5"	HDD	New	MaxDigitalData 8TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5inch Internal Hard Drive for NAS Network Storage (MD8000GSA25672NAS)
\$11.25	\$90	8 TB	3 years	Internal 3.5"	HDD	New	MaxDigitalData 8TB 7200 RPM 256MB Cache SATA 6.0Gb/s 3.5inch Internal Enterprise Hard Drive (MD8000GSA25672E) - 3 Years Warranty
\$11.87	\$190	16 TB		Internal 3.5"	HDD	New	16TB Exos X16 SATA 6Gb/s 7200RPM 3.5" Enterprise HDD — ST16000NM001G
\$13.33	\$80	6 TB	3 years	Internal 3.5"	HDD	New	MDD (MDD6TSATA6472DVR) 6TB 7200RPM 64MB Cache SATA 6.0Gb/s 3.5inch Internal Surveillance Hard Drive - 3 Years Warranty
\$13.33	\$40	3 TB		Internal	SAS	New	Seagate ST33000650SS Constellation ES.2 SAS 6Gb/s 3-TB Hard Drive

Attack 2: brute force attack

Mallory

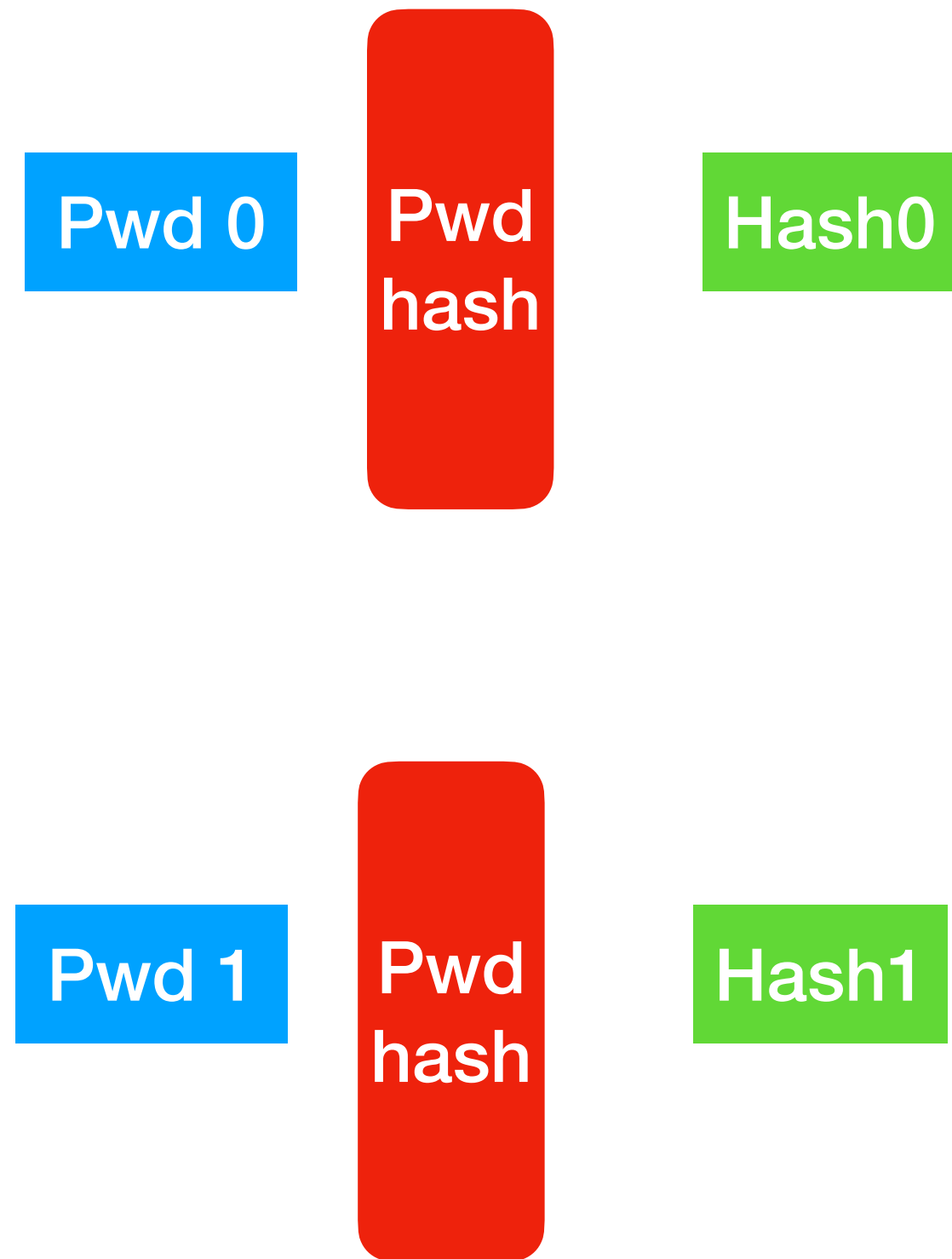
1. Buy storage system

- 2.

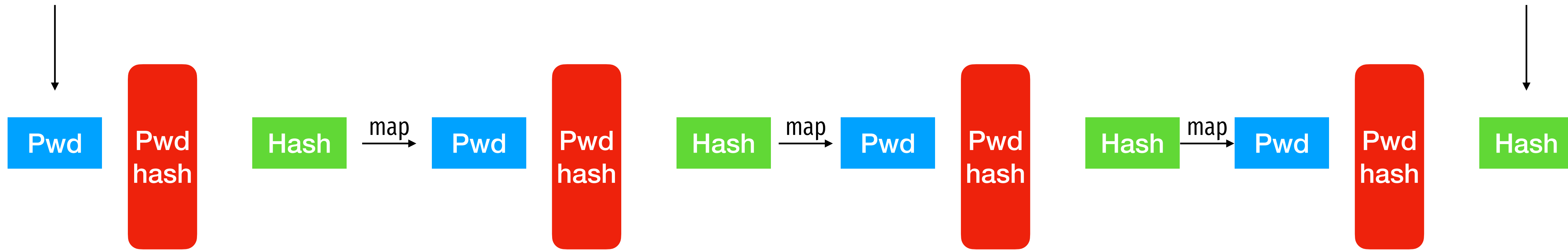
Bob



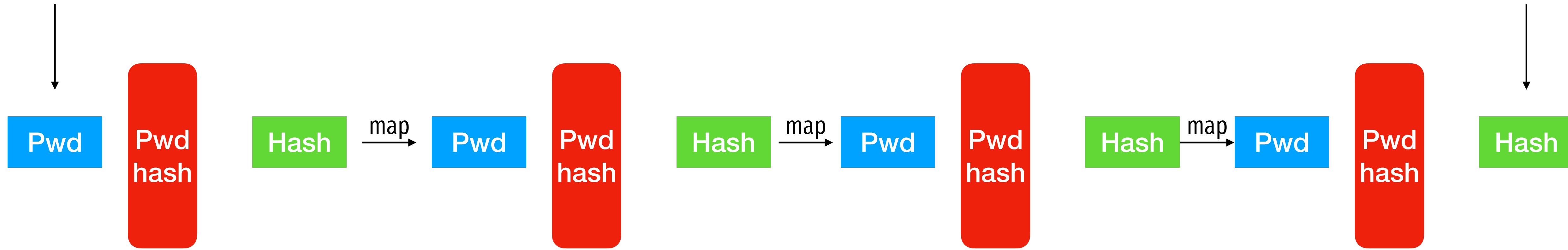
Classic Time-memory tradeoff



Classic Time-memory tradeoff



Classic Time-memory tradeoff



Example:

aaaaaa $\xrightarrow{\text{sha1}}$ f93...eae $\xrightarrow{\text{map}}$ sgyetr $\xrightarrow{\text{sha1}}$ b3f...bf8 $\xrightarrow{\text{map}}$ kiweuw $\xrightarrow{\text{sha1}}$

Only store first and last value in each row.



Given a hash [h] that you want to invert, you can:

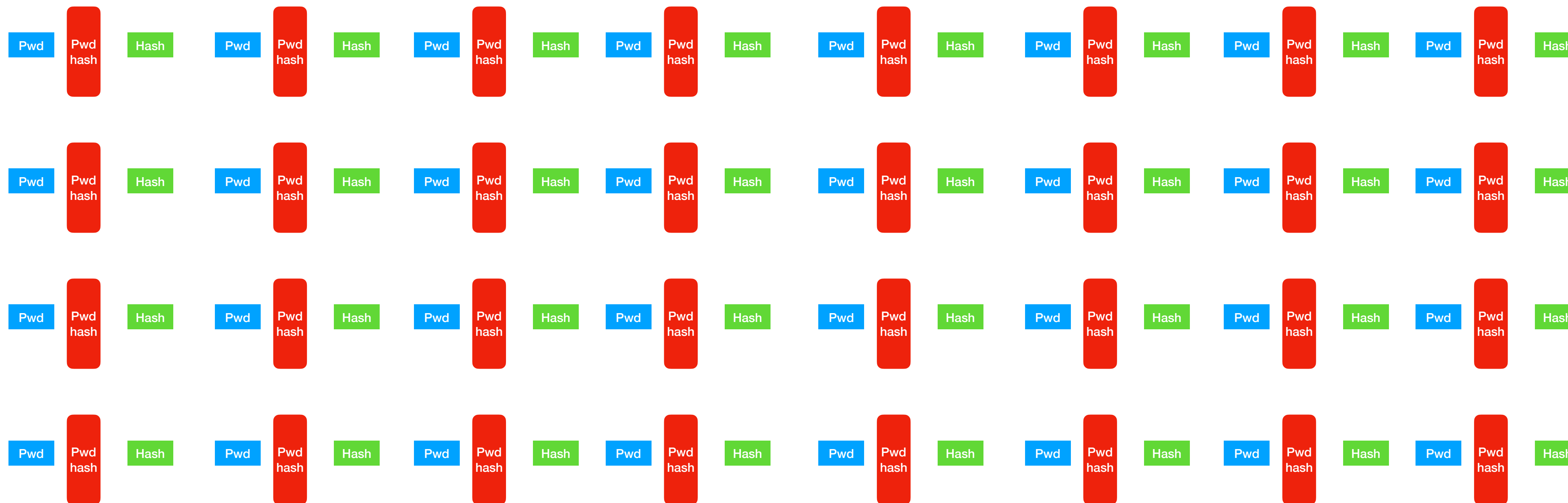
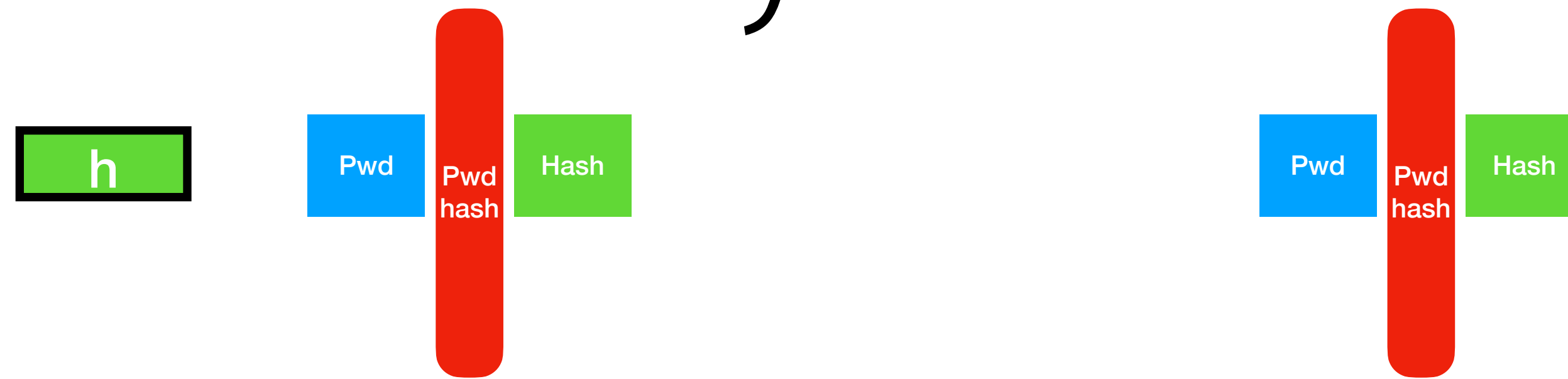
h



Example, suppose you want to invert **bf3...bf8**.

aaaaaa $\xrightarrow{\text{sha1}}$ **f93...eae** $\xrightarrow{\text{map}}$ **sgyetr** $\xrightarrow{\text{sha1}}$ **b3f...bf8** $\xrightarrow{\text{map}}$ **kiweuw** $\xrightarrow{\text{sha1}}$ **8ff...b24**

Classic Time-memory tradeoff



SHA1 Rainbow Tables

Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files	Performance
☰ sha1_ascii-32-95#1-7	ascii-32-95	1 to 7	70,576,641,626,495	99.9 %	52 GB 64 GB	Perfect Non-perfect	Perfect Non-perfect
☰ sha1_ascii-32-95#1-8	ascii-32-95	1 to 8	6,704,780,954,517,120	96.8 %	460 GB 576 GB	Perfect Non-perfect	Perfect Non-perfect
☰ sha1_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	221,919,451,578,090	99.9 %	127 GB 160 GB	Perfect Non-perfect	Perfect Non-perfect
☰ sha1_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	13,759,005,997,841,642	96.8 %	690 GB 864 GB	Perfect Non-perfect	Perfect Non-perfect
☰ sha1_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	104,461,669,716,084	99.9 %	65 GB 80 GB	Perfect Non-perfect	Perfect Non-perfect
☰ sha1_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	3,760,620,109,779,060	96.8 %	316 GB 396 GB	Perfect Non-perfect	Perfect Non-perfect

RainbowCrack Software Features

- High performance hash cracking on PC (> 10,000,000,000,000 plaintext tests per second)
- Optimized implementation of time-memory trade-off algorithm
- GPU acceleration with NVIDIA and AMD GPUs
- GPU acceleration with multiple GPUs
- Supports 64-bit Windows operating system
- Easy to use



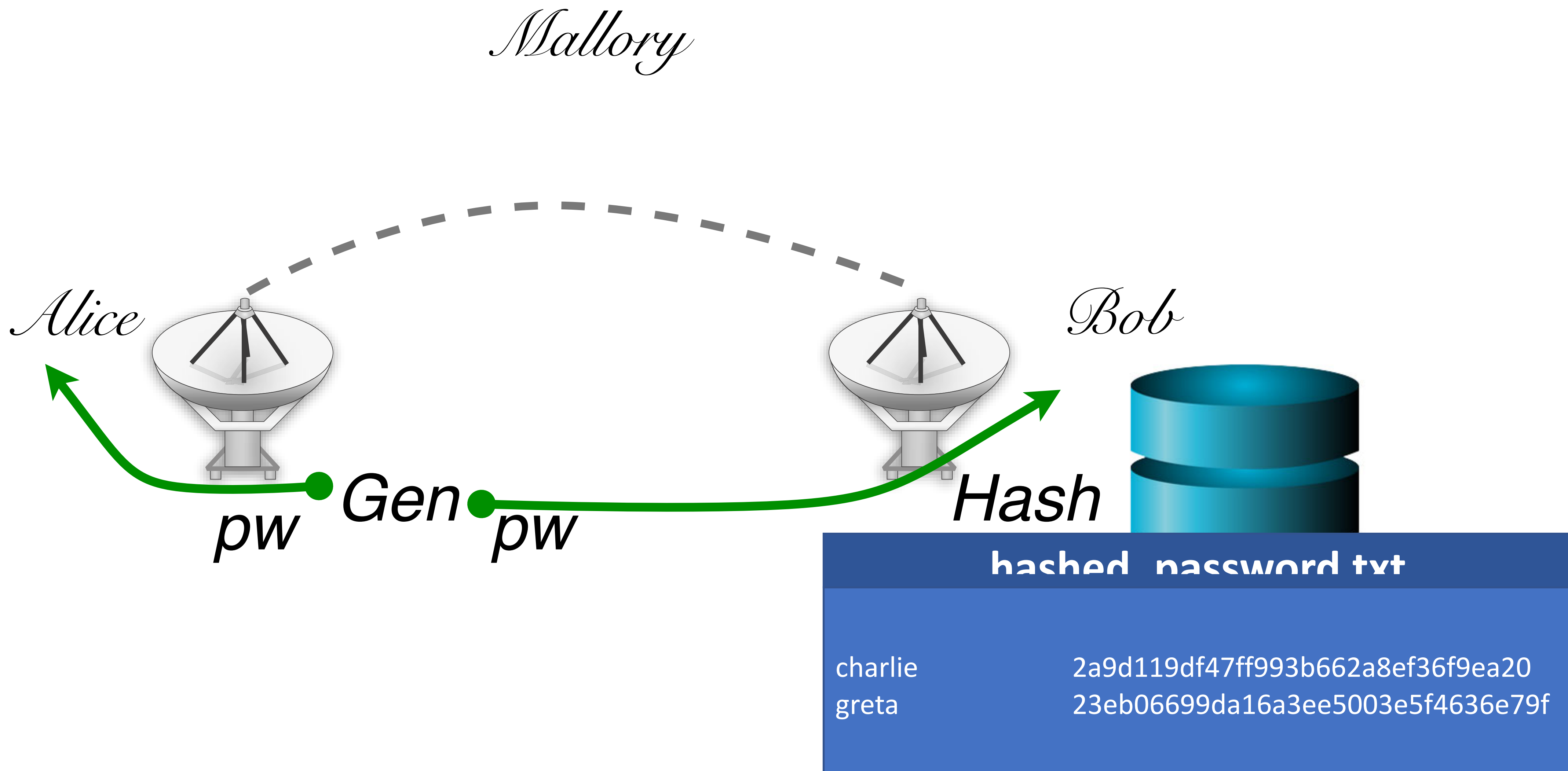
- [RainbowCrack 1.7 software](#)
- One [Seagate BarraCuda 6TB ST6000DM003 \(SATA\)](#) hard drive containing rainbow tables and software
- License in USB dongle



The attack is highly effective

<https://www.youtube.com/watch?v=TkMZJ3fTgrM>

Attack 2: offline brute force



How to hamper offline brute force attacks?

Mallory

hashed password.txt

charlie	2a9d119df47ff993b662a8ef36f9ea20
greta	23eb06699da16a3ee5003e5f4636e79f

Hardening Password Hashes

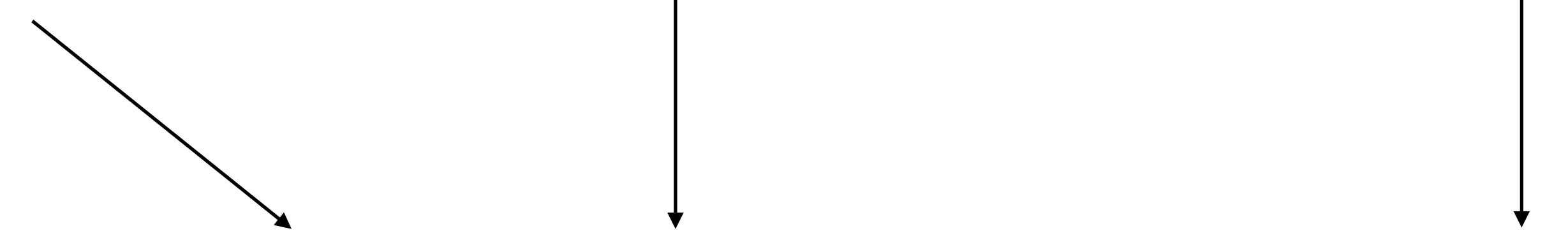
- **Key problem: cryptographic hashes are deterministic**
 - $\text{hash}(\text{'p4ssw0rd'}) = \text{hash}(\text{'p4ssw0rd'})$
 - This enables attackers to build lists of hashes
- **Solution: make each password hash unique**
 - Add a random **salt** to each password before hashing
 - $\text{hash}(\text{salt} + \text{password}) = \text{password hash}$
 - Each user has a unique, random salt
 - Salts can be stores in plain text

Example Salted Hashes

MD5 algorithm

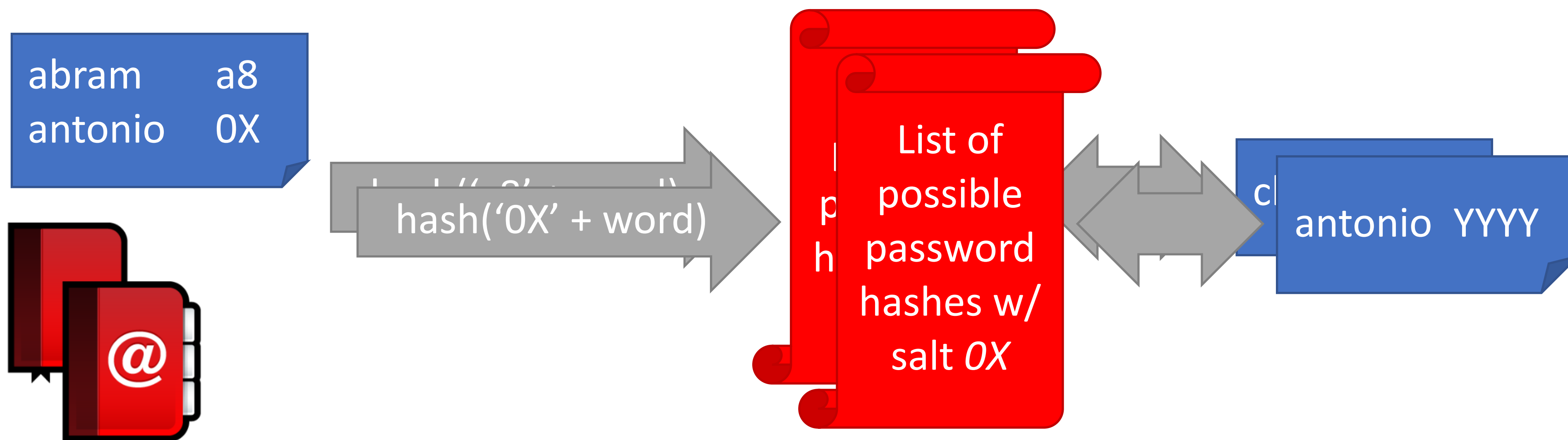
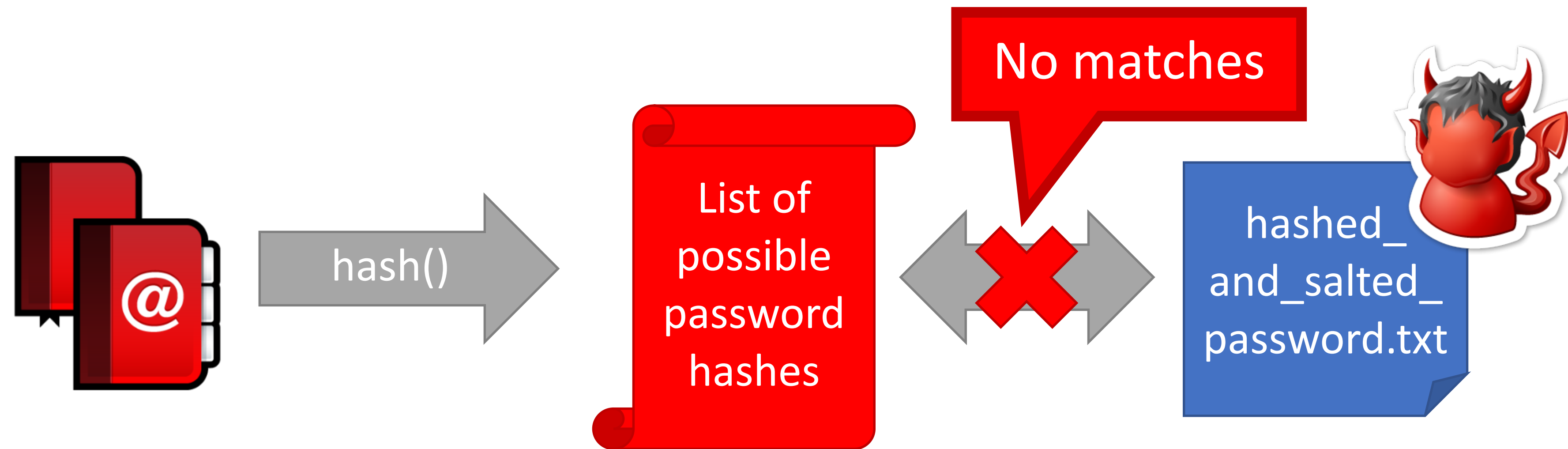
salt

hash



benvolio: \$1\$wSxbG4qj\$bm4SW0r5EG8KwxoIztctF0
abram: \$1\$oVoN6gZ1\$wZd8lNY0A7DGk7tSGhu3I/
antonio: \$1\$/nh0l9vD\$m04b0AIpaVjN0rvypUg9f.

Attacking Salted Passwords



Breaking Hashed Passwords

- **Stored passwords should always be salted**
 - Forces the attacker to brute-force each password individually
- **Problem: it is now possible to compute hashes very quickly**
 - GPU computing: hundreds of small CPU cores
 - nVidia GeForce GTX Titan Z: 5,760 cores
 - GPUs can be rented from the cloud very cheaply
 - \$0.9 per hour (2018 prices)

Examples of Hashing Speed

- A modern x86 server can hash all possible 6 character long passwords in 3.5 hours
 - Upper and lowercase letters, numbers, symbols
 - $(26+26+10+32)^6 = 690$ billion combinations
- A modern GPU can do the same thing in 16 minutes
- Most users use (slightly permuted) dictionary words, no symbols
 - Predictability makes cracking much faster
 - Lowercase + numbers $\rightarrow (26+10)^6 = 2B$ combinations

Hardening Salted Passwords

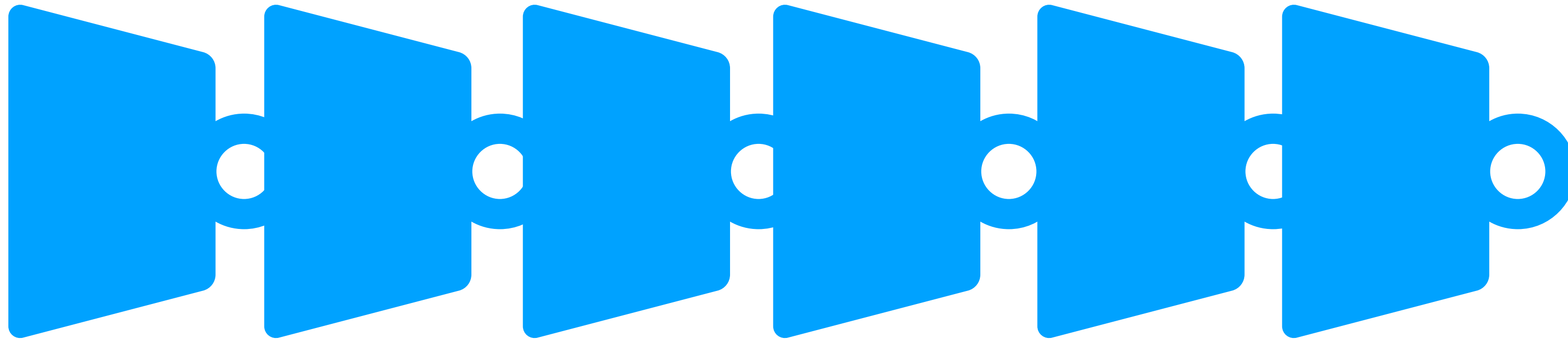
- **Problem:** typical hashing algorithms are too fast
 - Enables GPUs to brute-force passwords
- **Old solution:** hash the password multiple times
 - Known as **key stretching**
 - Example: *crypt* used 25 rounds of DES
- **New solution:** use hash functions that are designed to be **slow**
 - Examples: bcrypt, PBKDF2, scrypt
 - These algorithms include a **work factor** that increases the time complexity of the calculation
 - scrypt also requires a large amount of memory to compute, further complicating brute-force attacks

Slow hash movement



Iterated hash function {x times}

Pw
Salt



Hashed pwd

bcrypt Example

- Python example; install the *bcrypt* package

```
>>> import crypt
>>> bcrypt.hashpw(b'fooasdf', bcrypt.gensalt(8))
```



Work factor

Best practices so far:

Dealing With Breaches

- Suppose you build an extremely secure password storage system
 - All passwords are salted and hashed by a high-work factor function
- It is still possible for a dedicated attacker to steal and crack passwords
 - Given enough time and money, anything is possible
 - E.g. The NSA
- Question: is there a principled way to detect password breaches?

Honeywords

- Key idea: store multiple salted/hashed passwords for each user
 - As usual, users create a single password and use it to login
 - User is unaware that additional **honeywords** are stored with their account
- Implement a **honeyserver** that stores the index of the correct password for each user
 - Honeyserver is logically and physically separate from the password database
 - Silently checks that users are logging in with true passwords, not honeywords
- What happens after a data breach?
 - Attacker dumps the user/password database...
 - But the attacker doesn't know which passwords are honeywords
 - Attacker cracks all passwords and uses them to login to accounts
 - If the attacker logs-in with a honeyword, the honeyserver raises an alert!

Honeywords example



Bob

SHA512("fl" | "p4ssW0rd") → bHDJ8l



Cracked Passwords

User	PW 1	PW 2	PW 3
Bob	123456	p4ssW0rd	Turtles!
sandi	puppies	iloveyou	blizzard
Alice	coff33	3spr3ss0	qwerty

Database



User	Salt 1	H(PW 1)	Salt 2	H(PW 2)	Salt 3	H(PW 3)
Bob	aB	y4DvF7	fl	bHDJ8l	52	Puu2s7
sandi	0x	pIDS4F	K2	R/p3Y8	8W	S8x4Gk
Alice	9j	0F3g5H	/s	03d5jW	cV	1sRbJ5

Honeyserver



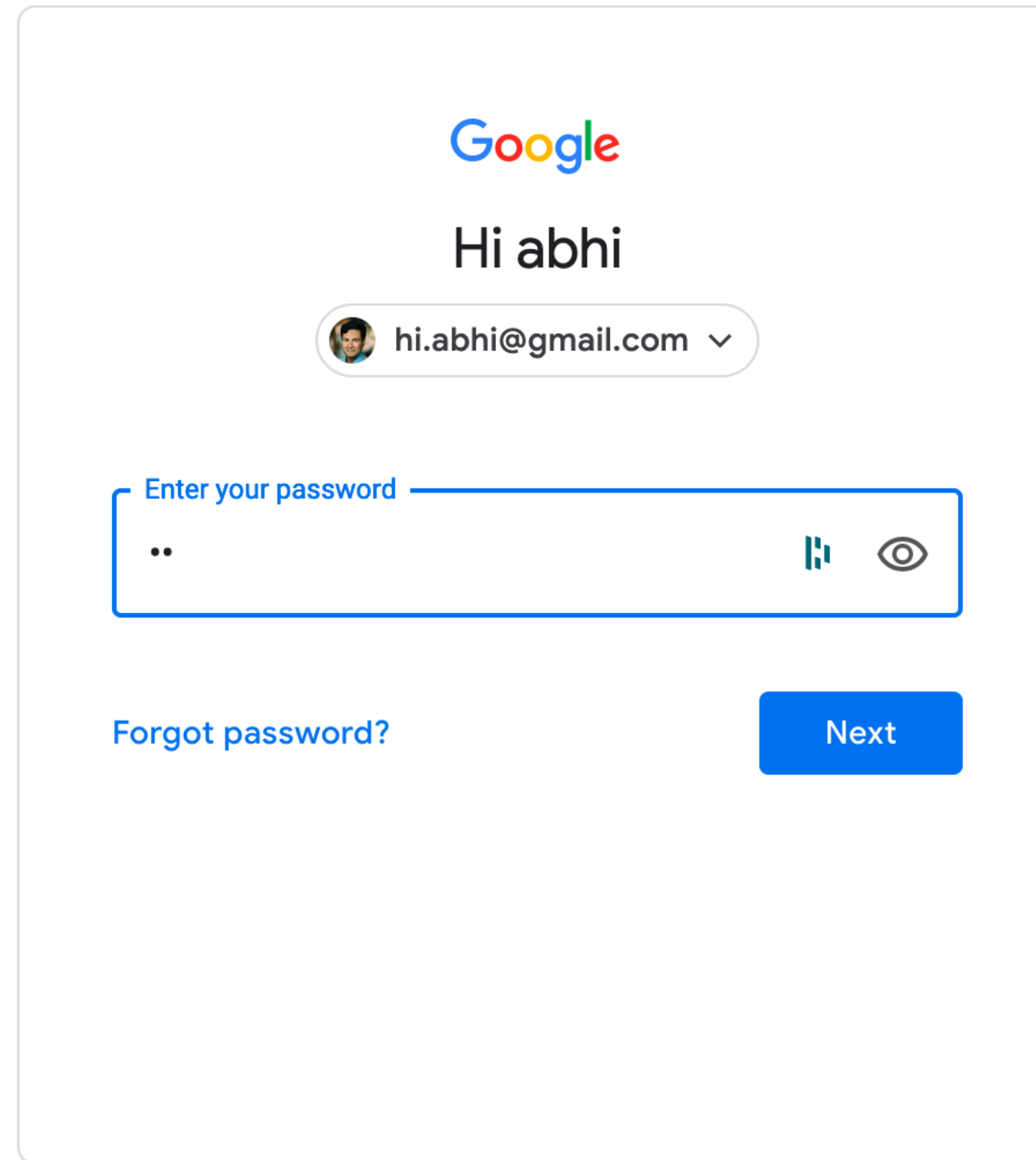
User	Index
Bob	2
sandi	3
Alice	1

Multiple layers of storage

Password Storage Summary

- 1. Never store passwords in plain text**
 - 2. Always salt and hash passwords before storing them**
 - 3. Use hash functions with a high work factor**
 - 4. Implement honeywords to detect breaches**
- These rules apply to any system that needs to authenticate users
 - Operating systems, websites, etc.

Still one problem?



The image shows a Google login interface. At the top is the Google logo. Below it, the text "Hi abhi" is displayed. Underneath is a dropdown menu showing a profile picture and the email address "hi.abhi@gmail.com" with a downward arrow. The main focus is on a password input field. The field has a blue border and a blue focus ring. The placeholder text "Enter your password" is at the top left of the field. Inside the field, there are two dots representing the password. To the right of the dots are two icons: a keyboard icon and an eye icon. Below the password field, there is a link "Forgot password?" on the left and a blue button labeled "Next" on the right.

Password Recovery/Reset

- Problem: hashed passwords cannot be recovered (hopefully)




“Hi... I forgot my password. Can you email me a copy? Kthxbye”

- This is why systems typically implement password **reset**
 - Use out-of-band info to authenticate the user
 - Overwrite `hash(old_pw)` with `hash(new_pw)`
- Be careful: its possible to crack password reset

Cracking Password Reset

- Typical implementations use **Knowledge Based Authentication (KBA)**
 - What was your mother's maiden name?
 - What was your prior street address?
 - Where did you go to elementary school
- **Problems?**
 - This information is widely available to anyone
 - Publicly accessible social network profiles
 - Background-check services like Spokeo
- **Experts recommend that services not use KBA**
 - When asked, users should generate random answers to these questions

Other roots of identity



Account recovery

hi.abhi@gmail.com ▾

Enter the last password you remember using with this Google Account

Enter last password

[Try another way](#)

Forgot username or password

Identification ▬ ▬ ▬ ▬

[Have a question? >](#)

Help us verify your identity.

For your security, please choose one of the options to verify your identity and provide the other requested information.

Choose one

Social Security number

[Don't have a Social Security number? >](#)

Account type Chase ATM/debit/prepaid card or credit card

Chase commercial loan

Other Chase account (e.g., checking, savings, mortgage application, commercial term loan, auto loan or lease)

Choosing Passwords

Bad Algorithms

Better Heuristics

Password Reuse

Password Reuse

- People have difficulty remembering >4 passwords
 - Thus, people tend to reuse passwords across services
 - What happens if any one of these services is compromised?
- Service-specific passwords are a beneficial form of compartmentalization
 - Limits the damage when one service is inevitably breached
- Use a password manager
- Some service providers now check for password reuse
 - Forbid users from selecting passwords that have appeared in leaks

Sites



Sort By: Folder (a-z)

Favorites (8)

AirBnB
fan@lastpass.comAmazon
fan@lastpass.com

Launch

Best Buy
fan@lastpass.comDropbox
fan@lastpass.comEvernote
fan@lastpass.comFacebook
fan@lastpass.comPocket
fan@lastpass.comTwitter
fan@lastpass.com

Banking and Finance (3)

Read Only • Shared Folder

Bank of America
fan@lastpass.comFidelity
fan@lastpass.comMint
fan@lastpass.com

Dashlane

The screenshot shows the Dashlane web interface. On the left is a dark sidebar with navigation options: VAULT (Passwords, Secure Notes, Personal Info, Payments, IDs, Receipts), SECURITY (Identity Dashboard, Password Health), and CONTACTS (Sharing Center, Emergency). At the bottom of the sidebar, it shows 'Getting Started' at 90% and '5 days left of Premium Extend Premium'. The main content area has a search bar, 'Add new' button, 'Password Changer', and 'Share' options. Below is a list of 22 passwords, each with a green database icon and details:

IP/ID	Username
1.1 (wgr614v5)	admin
1.20 (812b)	root
1.50 (airstation)	root
10.0.1.2	admin
10.0.1.50	812A12 (5Ghz)
10.250.224.2	ashelat
104.131.125.119	abhi@arqspin.com
11.1 (airstation)	root
11.1 (dd-wrt)	root

The screenshot shows the Mozilla website's 'Free password manager' page. The top navigation bar includes 'moz://a', 'Firefox Browsers', 'Products', 'Who We Are', 'Innovation', and a 'Get Mozilla VPN' button. The page title is 'Firefox Features → Free password manager'. The main content area features a large heading 'Free password manager' and a paragraph: 'Firefox securely stores your usernames and passwords for accessing websites, automatically fills them in for you the next time you visit a website, and lets you manage your stored logins with its built-in password management feature.' Below this, it says: 'With a [free Mozilla account](#) you can securely sync your passwords across all your devices. You can also access all of Mozilla's other privacy-respecting...'. The browser's address bar shows 'https://passwords.google.com'. Below the browser, the 'Google Account' section is visible, followed by the 'Password Manager' heading and a sub-heading: 'See, change, or remove passwords you saved in your Google Account. [Learn more](#)'. A 'Password Checkup' section is also present, with the text: 'Check your saved passwords to strengthen your security.' and a 'Go to Password Checkup' button.



Home

Notify me

Domain search

Who's been pwned

Passwords

API

About

Donate

';--have i been pwned?

Check if you have an account that has been compromised in a data breach

264

pwned websites

4,859,717,682

pwned accounts

61,081

pastes

59,268,789

paste accounts

Two Factor Authentication

Biometrics

SMS

Authentication Codes

Smartcards & Hardware Tokens

Types of Secrets

- Actors provide their secret to **log-in** to a system
- Three classes of secrets:
 1. Something you know
 - Example: a password
 2. Something you have
 - Examples: a smart card or smart phone
 3. Something you are
 - Examples: fingerprint, voice scan, iris scan

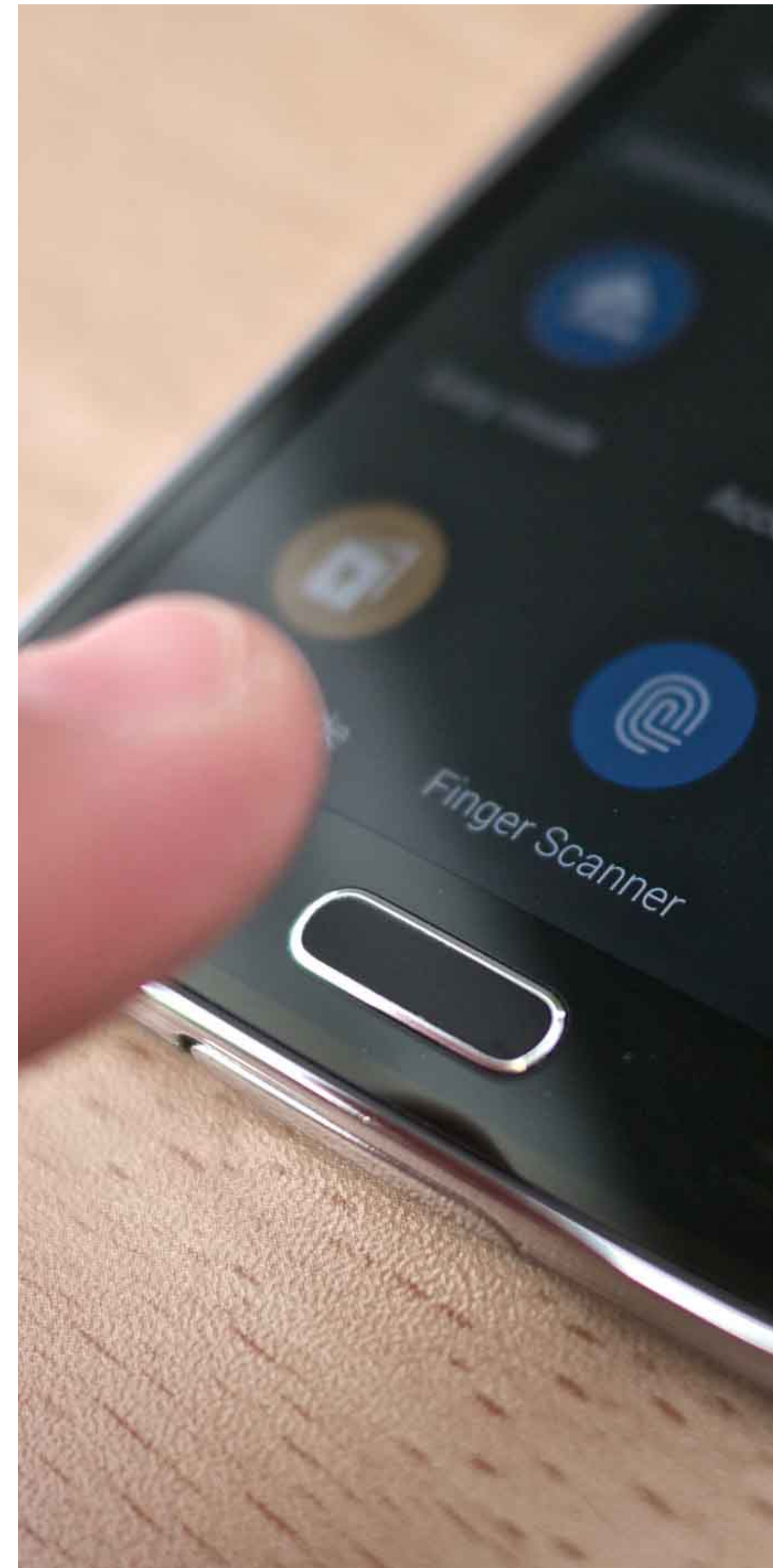
Biometrics

- ancient Greek: bios = "life", metron = "measure"
- Physical features
 - Fingerprints
 - Face recognition
 - Retinal and iris scans
 - Hand geometry
- Behavioral characteristics
 - Handwriting recognition
 - Voice recognition
 - Typing cadence
 - Gait

Fingerprints

- Ubiquitous on modern smartphones, some laptops
- Secure?
 - May be subpoenaed by law enforcement
 - Relatively easy to compromise
 1. Pick up a latent fingerprint (e.g. off a glass) using tape or glue
 2. Photograph and enhance the fingerprint
 3. Etch the print into gelatin backed by a conductor
 4. Profit ;)

https://www.theregister.co.uk/2002/05/16/gummi_bears_defeat_fingerprint_sensors/



Facial Recognition

- Popularized by FaceID on the iPhone X
- Secure?
 - It depends
- Vulnerable to law enforcement requests
- Using 2D images?
 - Not secure
 - Trivial to break with a photo of the target's face
- Using 2D images + 3D depth maps?
 - More secure, but not perfect
 - Can be broken by crafting a lifelike mask of the target





Specially processed area

2D images

Silicone nose

3D printed frame



By Press Association

Saturday, October 19, 2019 - 01:20 PM

Google has confirmed the Face Unlock system on its new Pixel 4 smartphone can allow access to the device even when the user has their eyes closed.

Early testers of the phone, as well as security experts, have raised concerns it could lead to unauthorised access to the device.

It has been suggested someone else could gain access to the phone by holding it in front of the face of its sleeping owner, but Google said it meets security requirements.

The technology giant unveiled the new phone earlier this week.

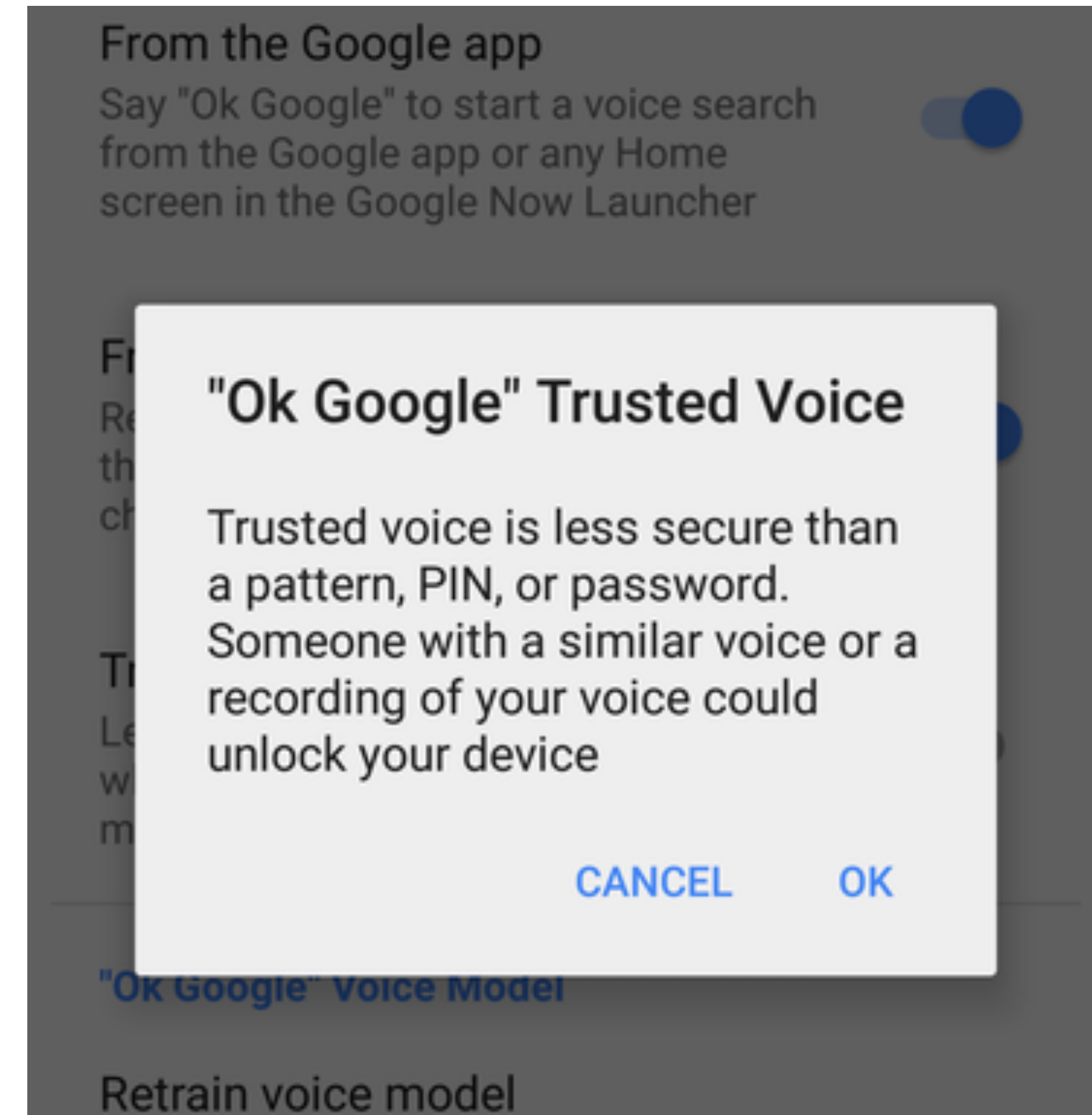
In a statement, Google said: "Pixel 4 Face Unlock meets the security requirements as a strong biometric and can be used for payments and app authentication, including banking apps.

"It is resilient against unlock attempts via other means, like with masks.

"If you want to temporarily disable Face Unlock, you can use lockdown mode to temporarily require a PIN/pattern/password.

Voice Recognition

- Secure?
 - Very much depends on the implementation
- Some systems ask you to record a static phrase
 - E.g. say “unlock” to unlock
 - This is wildly insecure
 - Attacker can record and replay your voice
- Others ask you to train a model of your voice
 - Train the system by speaking several sentences
 - To authenticate, speak several randomly chosen words
 - Not vulnerable to trivial replay attacks, but still vulnerable
 - Given enough samples of your voice, an attacker can train a synthetic voice AI that sounds just like you



Fundamental Issue With Biometrics

- Biometrics are immutable
 - You are the password, and you can't change
 - Unless you plan on undergoing plastic surgery?
- Once compromised, there is no reset
 - Passwords and tokens can be changed
- Example: the Office of Personnel Management (OPM) breach
 - US gov agency responsible for background checks
 - Had fingerprint records of all people with security clearance
 - Breached by China in 2015, all records stolen :(

Something You Have

- Two-factor authentication has become more commonplace
- Possible second factors:
 - SMS passcodes
 - Time-based one time passwords
 - Hardware tokens

SMS Two Factor

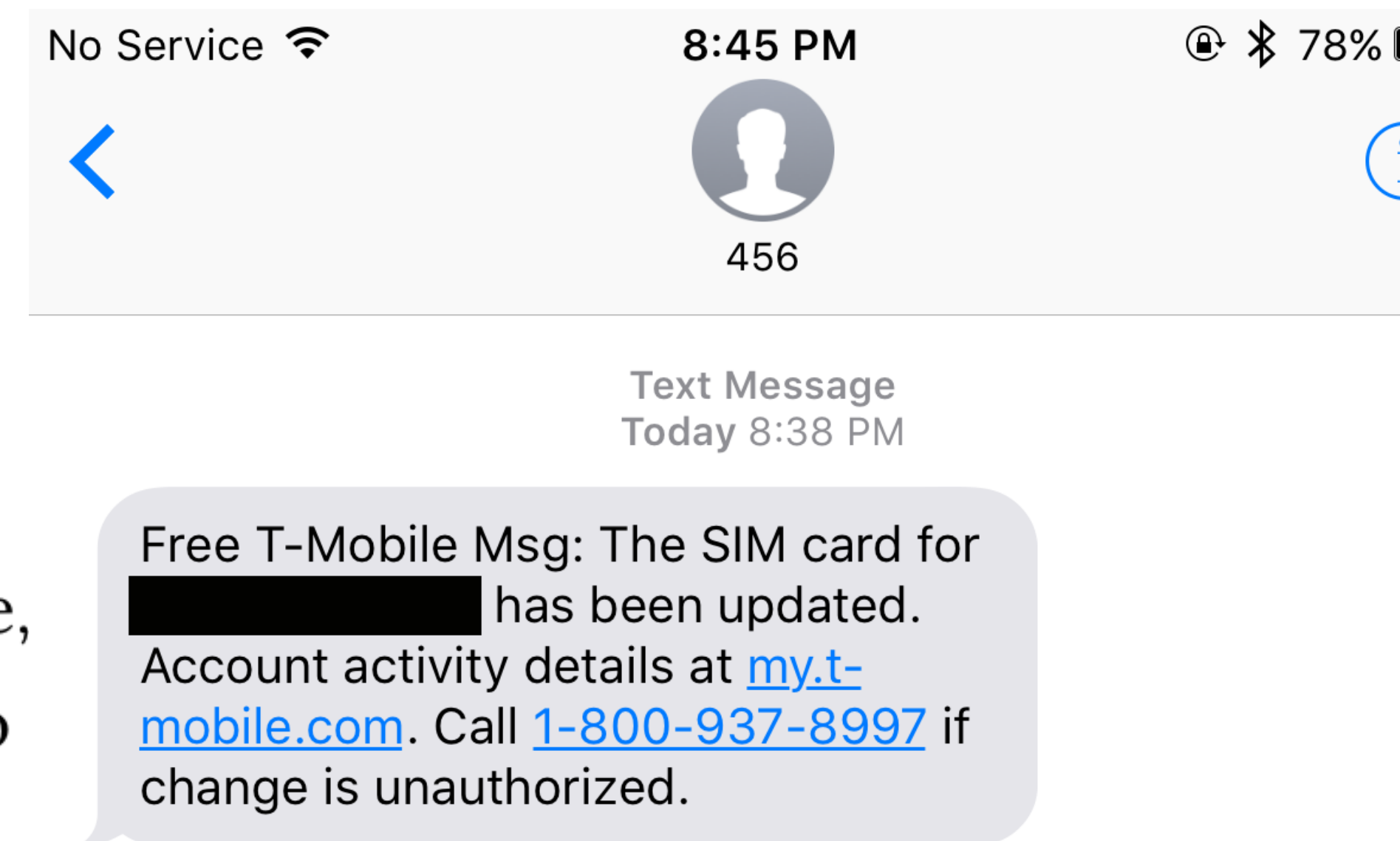
- Relies on your phone number as the second factor
 - Key assumption: only your phone should receive SMS sent to your number
- SMS two factor is deprecated. Why?
- Social engineering the phone company
 1. Call and pretend to be the victim
 2. Say “I got a new SIM, please activate it”
 3. If successful, phone calls and SMS are now sent to your SIM in your phone, instead of the victim
- Not hypothetical: successfully used against many victims



First, criminals call a cell phone carrier's tech support number pretending to be their target. They explain to the company's employee that they "lost" their SIM card, requesting their phone number be transferred, or ported, to a new SIM card that the hackers themselves already own. With a bit of social engineering—perhaps by providing the victim's Social Security Number or home address (which is often available from one of the many data breaches that have happened in the last few years)—the criminals convince the employee that they really are who they claim to be, at which point the employee ports the phone number to the new SIM card.

Game over.

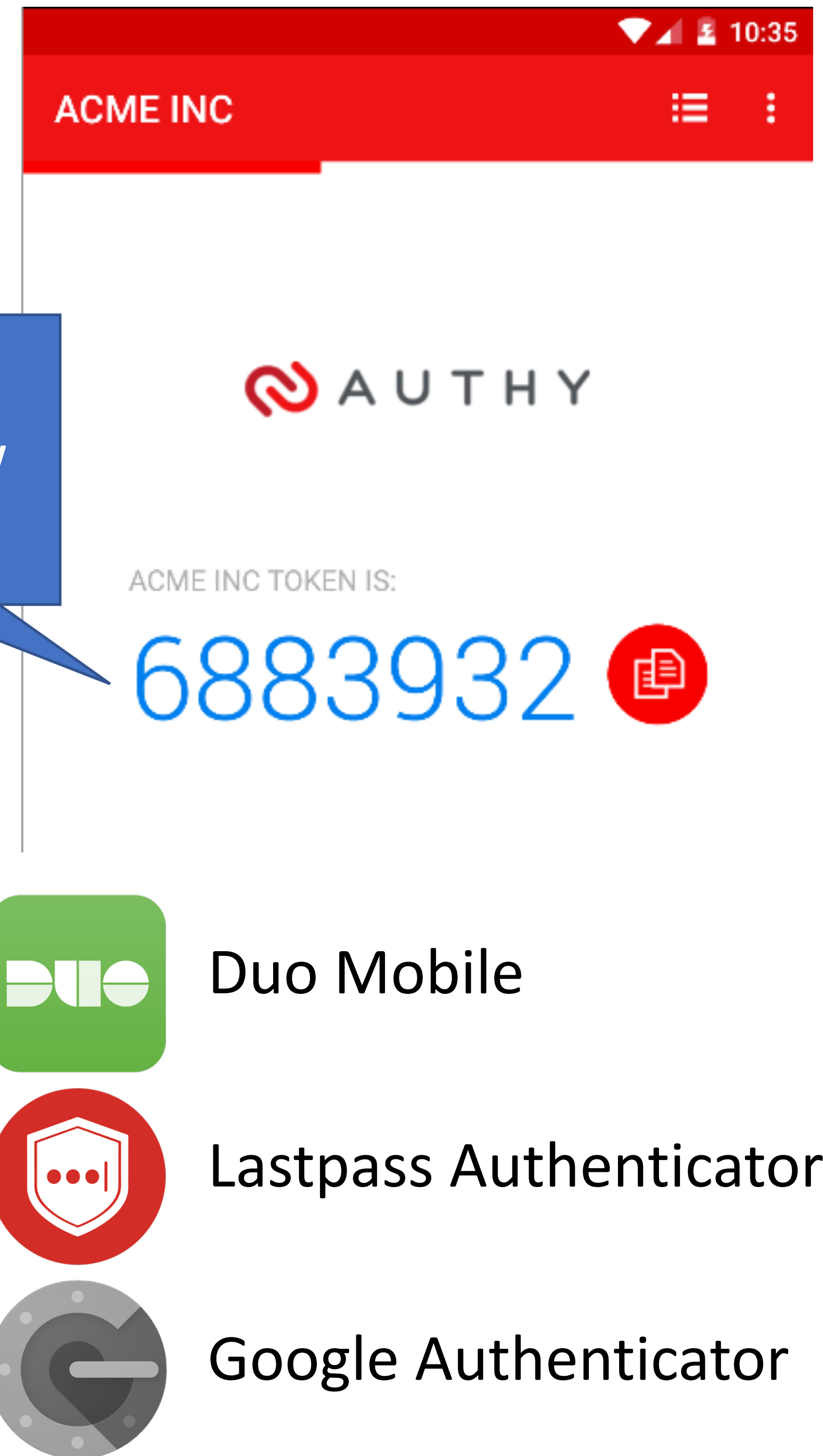
“With someone's phone number,” a hacker who does SIM swapping told me, “you can get into every account they own within minutes and they can't do anything about it.”



One Time Passwords

- Generate ephemeral passcodes that change over time
- To login, supply normal password and the current one time password
- Relies on a shared secret between your mobile device and the service provider
 - Shared secret allows both parties to know the current one time password

Changes every few minutes



Time-based One-time Password Algorithm

$T0$ = <the beginning of time, typically Thursday, 1 January 1970 UTC>

Tl = <length of time the password should be valid>

K = <shared secret key>

d = <the desired number of digits in the password>

$TC = \text{floor}((\text{unixtime}(\text{now}) - \text{unixtime}(T0)) / Tl),$

$\text{TOTP} = \text{HMAC}(K, TC) \% 10^d$

Specially formatted
SHA1-based signature

Given K , this algorithm can
be run on your phone and by
the service provider

Secret Sharing for TOTP

Enable Two-Step Sign in

An authenticator app generates the code automatically on your smartphone. Free apps are available for all smartphone platforms including iOS, Android, Blackberry and Windows. Look for an app that supports time-based one-time passwords (TOTP) such as Google Authenticator or Duo Mobile.

To set up your mobile app, add a new service and scan the QR code.



If you can't scan the code, enter this secret key manually: fvxo

[USE SMS INSTEAD](#)

[CANCEL](#)

[NEXT STEP](#)

[REFER A FRIEND](#)

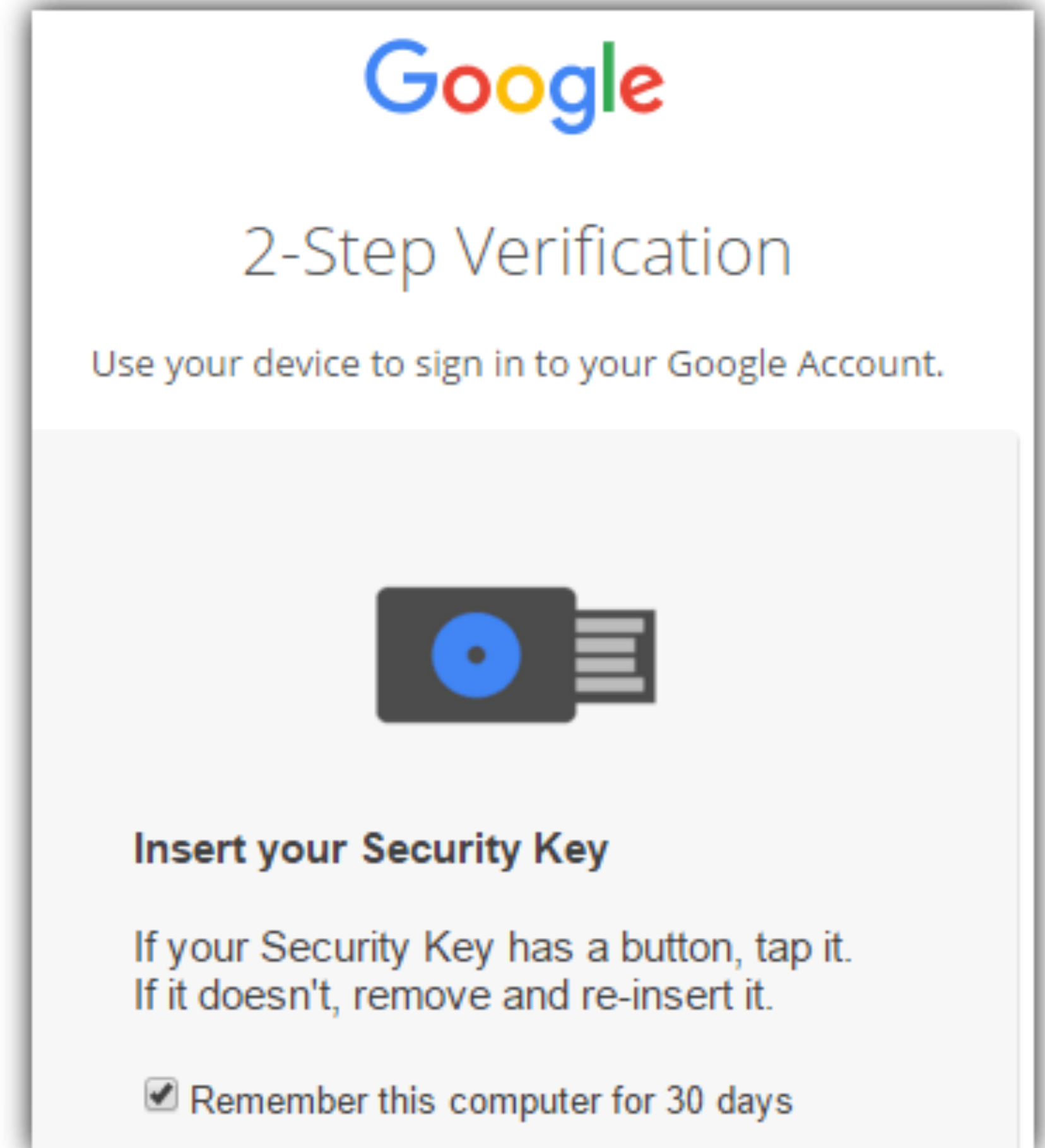
Hardware Two Factor

- Special hardware designed to hold cryptographic keys
- Physically resistant to key extraction attacks
 - E.g. scanning tunneling electron microscopes
- Uses:
 - 2nd factor for OS log-on
 - 2nd factor for some online services
 - Storage of PGP and SSH keys

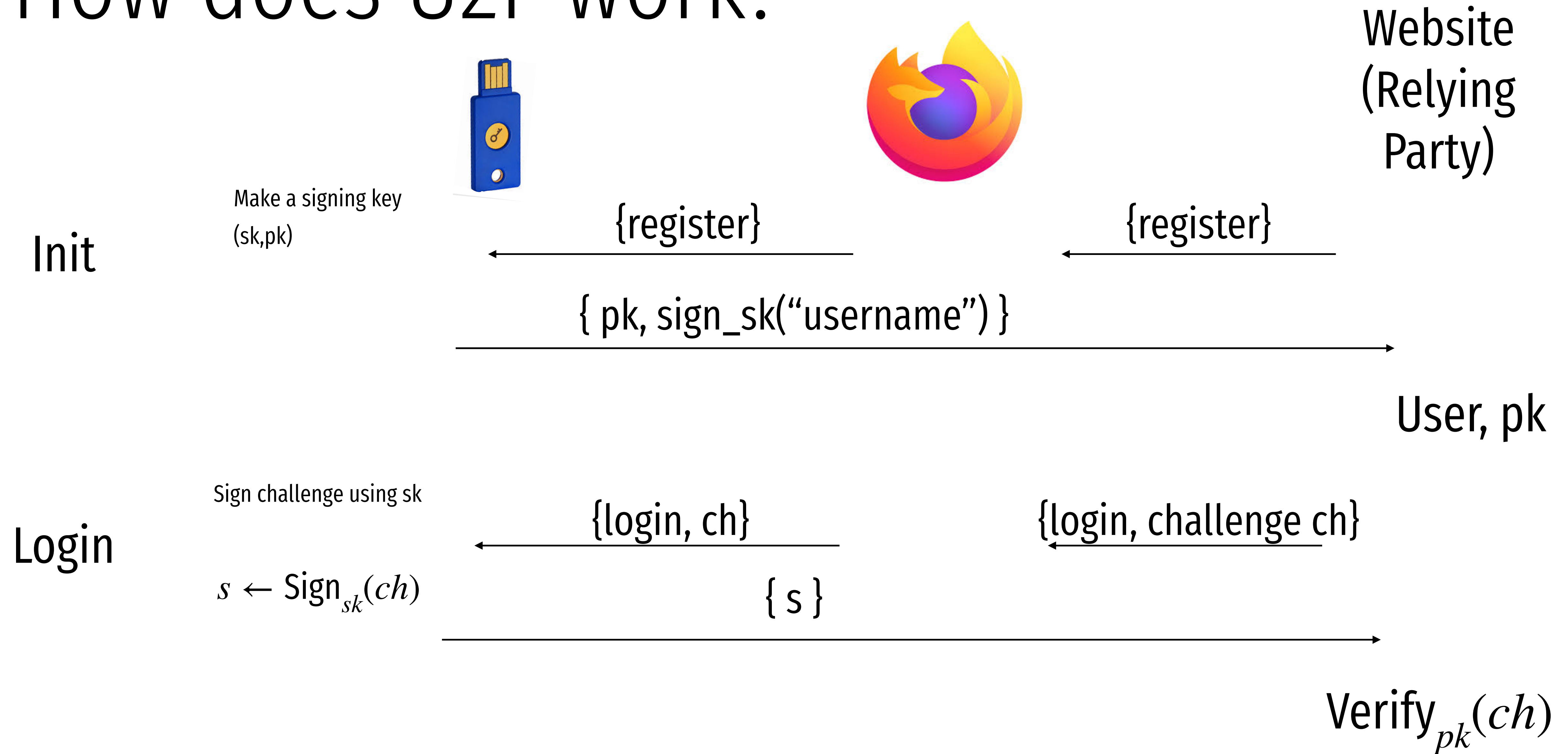


Universal 2nd Factor (U2F)

- Supported by Chrome, Opera, and Firefox (must be manually enabled)
- Works with Google, Dropbox, Facebook, Github, Gitlab, etc.
- Pro tip: always buy 2 security keys
 - Associate both with your accounts
 - Keep one locked in a safe, in case you lose your primary key ;)



How does U2F work?



Vulnerable to simple attack



Welcome

 hi.abhi@gmail.com ▾

Enter your password  

[Forgot password?](#)

[Next](#)

Simple Phishing

Lure: A spammed email with a call to action from a seemingly legitimate source encouraging the user to visit a hook website.

Hook: A website designed to mimic legitimate site and collect confidential information.