User Authentication

Vítor Francisco Fonte, vff@di.uminho.pt, University of Minho, 2025

Overview

- Password-Based Authentication
    - Attacks and defences
- Approaches to User Authentication
    - Something you know, have, are, or do, and where you are
    - Multi factor authentication
    - Single Sign-On

User Authentication

- **Why user authentication?**
    - Identity is a parameter of an **access control** decision
    - Identity is recorded when **logging** security-relevant events to an **audit trail**
- **User authentication**, helps ensure**:**
    - Only system users have access to system resources (first line of defence from outside)
    - A system user cannot impersonate another, gaining access to unauthorised resources
    - Contributes to security auditing and resource accountability

User Authentication

- Two step process
    1. **Identification**: you announce who you are
    2. **Authentication**: you prove you are who you claim to be
- Users:
    - **Humans and non-humans (virtual)**
    - Often part of groups
    - Users and groups are considered to be Principals in Access Control

Password-Based Authentication

Most common form of authentication

1. The user inputs a name and a password
2. The system verifies the user input against entries stored in a password file
3. If the verification is successful, a user authenticated session is established:
    - The system may require further authentication at a later stage (e.g. idle time, sensitive operation)
    - The system may also choose to simply close a session automatically if it is idle for too long
4. Otherwise:
    - Usually, the login screen will be displayed again and the user will be able to start your next attempt
    - The system may prevent or delay further attempts when reached threshold of failed authentications

Password-Based Authentication

Overview of attack strategies

- Intercept the password at the moment it is first set or reset (paper, e-mail)
- Get the password by guessing it
- Get the password through phishing or spoofing, or by key-loggers
- Get the password from the system by compromising the password file
- Get the password from by other social engineering techniques

Password-Based Authentication

Password guessing: strategies and techniques

- **Exhaustive search (brute force):**
  - All possible combinations of valid symbols, up to a certain length
- **Intelligent search (restricted name space):**
  - Passwords somehow associated with the user (name, friends, relatives, OSINT)
  - Passwords that can be traced to the user from compromised credentials (leaks)
  - Passwords that are known to be popular, from books, dictionaries, etc (wordlists)
  - Combine all of the above using popular password patterns (heuristics)

Password-Based Authentication

Password guessing: defences

- **Change default passwords:** when systems are delivered, they often come with default accounts such as ‘system’ with default password ‘manager’. This helps the field engineer to install the system, but if the password is left unchanged the attacker has an easy job getting into the system. In this particular example, the attacker even gets access to a privileged account.

- **Password length:** to thwart exhaustive search, a minimal password length should be prescribed. Standard Unix systems, however, have a maximal password length set to eight characters only.

- **Password format:** mix upper and lower case symbols and include numerical and other non-alphabetical symbols in your password. The size of the password space is at least |A|n where n is the minimal password length and |A| is the size of the character set used for constructing passwords.

- **Avoid obvious passwords:** do not be surprised to find out that attackers are equipped with lists of popular passwords and be aware that dictionary attacks have extended the scope of ‘obvious’ quite substantially. Today, you can find an on-line dictionary for almost every language.

Password-Based Authentication

Password guessing: help from the system

- **Password checkers:** as a system manager, you can use tools that check passwords against some dictionary of ‘weak’ passwords and prevent users from choosing such passwords. This imitates – and pre-empts – dictionary attacks against the system.
- **Password generation:** some operating systems include password generators producing random but pronounceable passwords. Users are not allowed to pick their own password but have to adopt a password proposed by the system.
- **Password ageing:** an expiry date for passwords is set, forcing users to change passwords at regular intervals. There may be additional mechanisms to prevent users from choosing previous passwords, e.g. a list of the last ten passwords used. Still, determined users will be able to revert to their favourite password by making a sufficient number of changes until their old password is accepted again.
- **Limit login attempts:** the system monitors unsuccessful login attempts and reacts by locking the user account completely or at least for a certain period of time to prevent or discourage further attempts. The time the account is locked could be increased in proportion to the number of failed attempts.

Password-Based Authentication

Password guessing: the human factor

- **Are longer, complex password better?**
  - Users are unlikely to memorised such a password
  - So they tend to write it down somewhere (in a place close to the computer)
  - Passwords more accessible to attackers with physical access to the computer
- **What about frequent, mandatory changing of passwords?**
  - Users often resort to password patterns based on counters or dates
  - Passwords may end up being more easily guessable
- **Obtrusive security mechanisms often lead to weaker system security!**

Password-Based Authentication

Password: lessons from experience

- **People are best at memorising passwords they use regularly**
  - Passwords work reasonably in contexts they are used frequently
  - But not with systems used only occasionally
- **Advice on changing passwords:**
  - Immediately type the new password several times
  - Avoid doing it before weekends or holidays (or academic festivities 😉)

Password-Based Authentication

Wordlists: line-oriented text files

- **Common Passwords**: Lists of frequently used passwords compiled from multiple data breaches. These can be used in brute-force attacks, to build rainbow tables, and in conjunction with algorithms to conduct more advanced password brute-force enumeration techniques.
- **Dictionary Words**: Words from dictionaries of various languages, often used in dictionary attacks to guess passwords based on real words.
- **Popular References**: Words associated with historical or fictional characters, celebrities, birthdays, aniversaries of events, etc.
- **Passphrases**: Combinations of words or phrases that might be used in more complex passwords.
- **Customised Lists**: Wordlists tailored for specific targets, containing info relevant to the target's environment, such as local language, cultural references, or industry-specific terms.

Password-Based Authentication

Wordlists: what else are they used for?

- **Building Rainbow Tables**: Wordlists are used to generate rainbow tables, which are precomputed tables used to reverse cryptographic hash functions, making it easier to crack hashed passwords.
- **Directory and File Enumeration in Reconnaissance and Penetration Testing**: During reconnaissance and penetration testing, wordlists are employed to guess directory and filenames on web servers, aiding in the discovery of hidden resources and potential vulnerabilities.

Password-Based Authentication

Password guessing: credential stuffing

- Attackers collect credentials from data breaches
  - **User names and or email addresses** and the corresponding password
  - **Passwords** already **in clear text**
- Used in **automated credential injection attacks** against web applications
  - To gain unauthorised access to user accounts
- Compromised passwords can also be incorporated in wordlists

Password-Based Authentication

Examples of wordlists and compromised credentials

- **SecLists' wordlist collection (2018):**
  - https://github.com/danielmiessler/SecLists/tree/master/Passwords
- **RockYou2024 (2024)**:
  - Almost 10 billion compromised credentials
  - "Search and you shall find", of course
  - But always consider possible legal implications
- **Remember all topics and resources are to be used exclusively for studying.**

Password-Based Authentication

Spoofing, Phishing and Social Engineering

- **Password-based authentication is unilateral (common case):**
    - The system verifies the credentials presented by the user
    - But the user often does not know who has received the credentials
- **Phishing and spoofing attacks (also related to social engineering):**
    - The user voluntarily sends the password over a channel
    - But is misled about the end point of the channel
- **In social engineering attacks (in general):**
    - The user can be tricked or compelled to share the credentials

Password-Based Authentication

Spoofing attacks

- **The attacker presents the user with a fake login procedure.**
- Example: A program runs a **fake login screen** in an idle computer that records the credentials entered by a user, displays a failed authentication message that the user takes as its fault, and exits to the actual system login screen.
- Example: A **fake version of a website** the user is (somehow) directed to (e.g. through a malicious link).

Password-Based Authentication

Spoofing attacks: defences

- **Number of failed logins**: Displaying the number of failed logins may indicate to the user that an attack has happened.
- **Trusted path**: a guarantee that the user communicates with the actual system and not with a spoofing program (e.g CTRL-ALT-DEL cannot be handled by user programs and invokes the Microsoft Windows OS login screen).
- **Mutual authentication**: a system may be required to authenticate itself to the user (e.g. a valid certificate issued by a trusted authority in HTTPS).

Password-Based Authentication

Phishing attacks

- Users should take care to enter their passwords only at the ‘right’ site, but in practice it is **not always easy to recognise an attack**.
- Example: A message could claim to come from a service you are using, telling you about an upgrade of security procedures, and asking you to enter your username and password at the new security site.

Password-Based Authentication

Social engineering attacks

- The attacker may impersonate a user and trick a system operator into releasing the password to the attacker.
- Social engineering attacks are more successful when they better understand the psyche of the target (via personal knowledge or OSINT).
  - Is this a person that can be bullied?
  - Is this a person that is very supportive of struggling users?
- Crucial: security training, policies, practices and regular audit of organisations

Password-Based Authentication

Password “caching” attacks

- **Passwords are held temporarily in intermediate storage** locations such as buffers, caches, or even a web page. The management of these storage locations is normally beyond the control of the user and a password may be kept longer than the user expects.
- Example: A user of a web browser may be able to recover a previously terminated authentication session by clicking the back button, sometimes even when the session was terminated by that user. Sometimes the user credentials themselves can be recovered using the same technique.
- Example: User credentials may be exposed in temporary files, in the space that once belonged to a now deleted file, in the volatile memory chips (RAM), due to buffer overruns, inadequate use or improper implementation of cryptographic protocols, etc.

Password-Based Authentication

Protecting the password file

- An attacker can directly **impersonate** a user when the contents of an **unencrypted password file** are disclosed or when entries in the password file can be modified.
- Even the **disclosure of encrypted passwords** may be a concern**. Offline dictionary attacks** can then be conducted and protection measures such as limiting the number of unsuccessful login attempts would not come into play.

Password-Based Authentication

Protecting the password file: approaches

- **Cryptographic protection:**
  - Mitigate the risk of guessing attacks
- **Access control enforced by the operating system**:
  - Mitigate the risk of unauthorised access to the encrypted passwords
- **Slow down password verification:**
  - Further mitigate the risk of guessing attacks

Password-Based Authentication

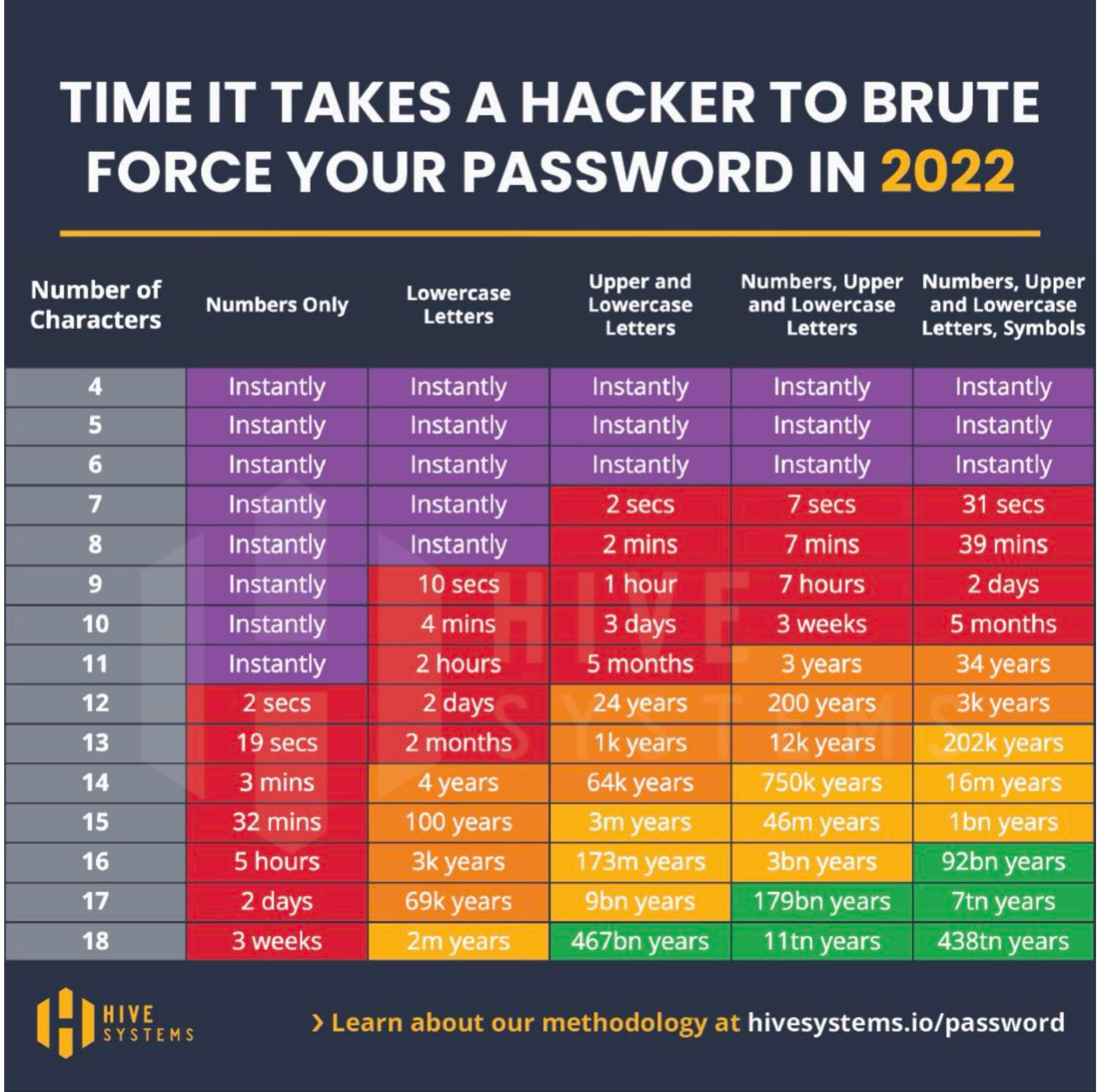
Protecting the password file: cryptographic protection

- **No need for an encryption** algorithm. A **one-way function** will do the job.
    - A one-way function is a function that is relatively easy to compute but significantly harder to undo or reverse. That is, given x it is easy to compute *f(x)*, but given *f(x)* it is hard to compute x.
- **Instead of the password *x*, the value *f(x)* is stored in the password file.**
    - When a user logs in and enters a password, say *x′*, the system applies the one-way function *f* and then compares *f(x′)* with the expected value *f(x)*. If the values match, the user has been successfully authenticated. If *f* is a proper one-way function, it is not feasible to reconstruct a password *x* from *f(x)*.

Password-Based Authentication

Protecting the password file: cryptographic protection

- The password file could now be left world-readable except for **offline dictionary attacks**.
  - The attacker hashes all words in a dictionary and compares the results against the hashed entries in the password file.
  - If a match is found, the attacker knows that user’s password.
- **Key stretching**: one-way functions can be repeatedly applied to slow down dictionary attacks



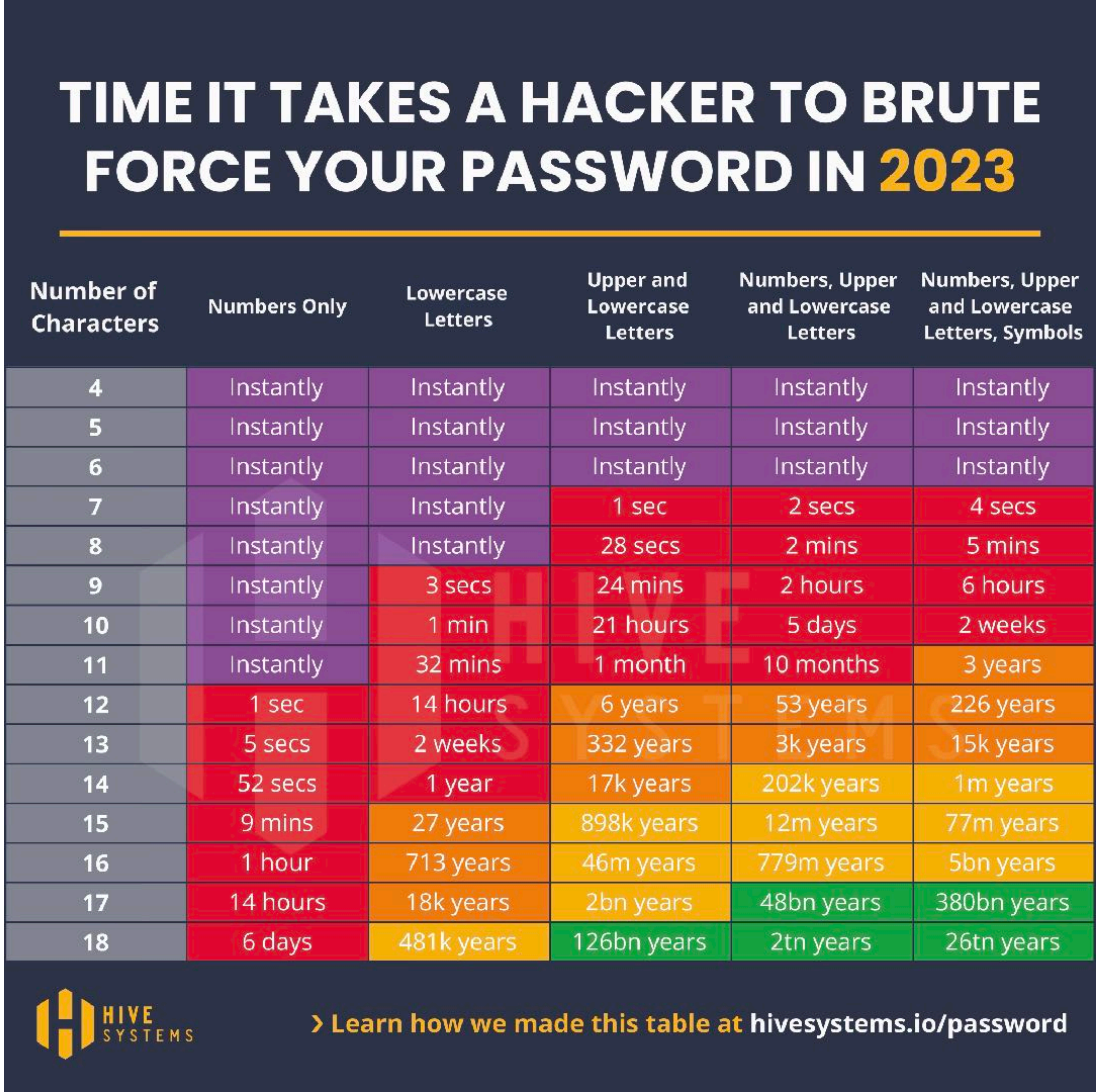
TIME IT TAKES A HACKER TO BRUTE FORCE YOUR PASSWORD IN 2022

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	Instantly	Instantly
7	Instantly	Instantly	2 secs	7 secs	31 secs
8	Instantly	Instantly	2 mins	7 mins	39 mins
9	Instantly	10 secs	1 hour	7 hours	2 days
10	Instantly	4 mins	3 days	3 weeks	5 months
11	Instantly	2 hours	5 months	3 years	34 years
12	2 secs	2 days	24 years	200 years	3k years
13	19 secs	2 months	1k years	12k years	202k years
14	3 mins	4 years	64k years	750k years	16m years
15	32 mins	100 years	3m years	46m years	1bn years
16	5 hours	3k years	173m years	3bn years	92bn years
17	2 days	69k years	9bn years	179bn years	7tn years
18	3 weeks	2m years	467bn years	11tn years	438tn years

HIVE SYSTEMS

> Learn about our methodology at hivesystems.io/password

hardware: 8 x A100
password hash: MD5



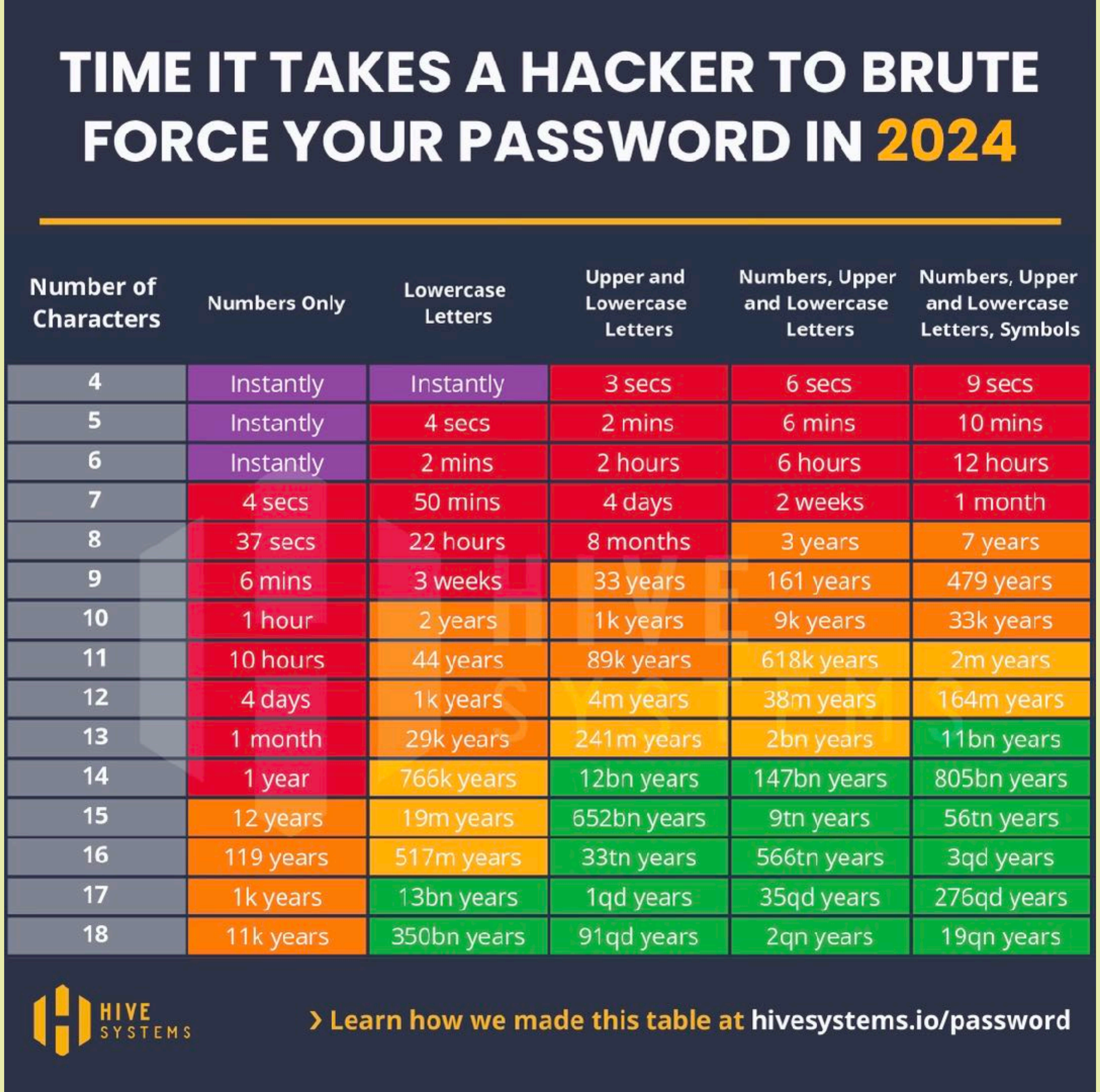
TIME IT TAKES A HACKER TO BRUTE FORCE YOUR PASSWORD IN 2023

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	Instantly	Instantly	Instantly
5	Instantly	Instantly	Instantly	Instantly	Instantly
6	Instantly	Instantly	Instantly	Instantly	Instantly
7	Instantly	Instantly	1 sec	2 secs	4 secs
8	Instantly	Instantly	28 secs	2 mins	5 mins
9	Instantly	3 secs	24 mins	2 hours	6 hours
10	Instantly	1 min	21 hours	5 days	2 weeks
11	Instantly	32 mins	1 month	10 months	3 years
12	1 sec	14 hours	6 years	53 years	226 years
13	5 secs	2 weeks	332 years	3k years	15k years
14	52 secs	1 year	17k years	202k years	1m years
15	9 mins	27 years	898k years	12m years	77m years
16	1 hour	713 years	46m years	779m years	5bn years
17	14 hours	18k years	2bn years	48bn years	380bn years
18	6 days	481k years	126bn years	2tn years	26tn years

HIVE SYSTEMS

> Learn how we made this table at hivesystems.io/password

hardware: 8 x RTX 4090
password hash: MD5



TIME IT TAKES A HACKER TO BRUTE FORCE YOUR PASSWORD IN 2024

Number of Characters	Numbers Only	Lowercase Letters	Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters	Numbers, Upper and Lowercase Letters, Symbols
4	Instantly	Instantly	3 secs	6 secs	9 secs
5	Instantly	4 secs	2 mins	6 mins	10 mins
6	Instantly	2 mins	2 hours	6 hours	12 hours
7	4 secs	50 mins	4 days	2 weeks	1 month
8	37 secs	22 hours	8 months	3 years	7 years
9	6 mins	3 weeks	33 years	161 years	479 years
10	1 hour	2 years	1k years	9k years	33k years
11	10 hours	44 years	89k years	618k years	2m years
12	4 days	1k years	4m years	38m years	164m years
13	1 month	29k years	241m years	2bn years	11bn years
14	1 year	766k years	12bn years	147bn years	805bn years
15	12 years	19m years	652bn years	9tn years	56tn years
16	119 years	517m years	33tn years	566tn years	3qd years
17	1k years	13bn years	1qd years	35qd years	276qd years
18	11k years	350bn years	91qd years	2qn years	19qn years

HIVE SYSTEMS

› Learn how we made this table at hivesystems.io/password

hardware: 8 x RTX 4090
password hash: bcrypt

Password-Based Authentication

Protecting the password file: enforcing access control

- **Access control mechanisms** in the operating system **restrict access to files** and other resources to users holding the appropriate privileges.
- **Only privileged users may have write access to the password file**. Otherwise, an attacker could get access to the data of other users simply by changing their password, even if it is protected by cryptographic means.
- If read access is restricted to privileged users, passwords in theory could be stored unencrypted.
- But if the **password file contains information that is also required by unprivileged users**, then the password file must contain encrypted passwords. However, such a file can still be used in dictionary attacks.
- A typical example is /etc/passwd in Unix. Therefore, many versions of Unix store enciphered passwords in a file that is not publicly accessible. Such files are called **shadow password files**.

Password-Based Authentication

Protecting the password file: enforcing access control in Linux

- The password file **/etc/passwd** contains no passwords (clear-text or otherwise)
  - Hashes of (salted) passwords are stored in **/etc/shadow**
- **The root user:**
  - Can read and write all those files
- **Regular users:**
  - Can read /etc/passwd but not /etc/shadow
  - Cannot write any of those files
- So… how can regular users change theirs passwords? ;-)

Password-Based Authentication

Protecting the password file: password salting

- When a password is encrypted for storage, additional information, the **salt**, is appended to the password **before hashing**. The salt is then stored with the hashed password.
- If two users have the same password, they will therefore have different entries in the file of encrypted passwords.
- **Salting slows down dictionary attacks** even further as it is no longer possible to search for the passwords of several users simultaneously.
- Salting makes **impractical the usage of pre-calculated hashed passwords** (**Rainbow Tables**).

Approaches to User Authentication

- Something you know
- Something you hold (or have)
- Who you are
- What you do
- Where you are

Approaches to User Authentication

Something you know

- **The user has to know some ‘secret’** to be authenticated
  - The secret had to be previously shared with the system
- Examples:
  - A password when a user logs in
  - A PIN when someone uses a bank card
  - Personal information in when calling customer support

Approaches to User Authentication

Something you know

- **Anybody who obtains your secret ‘is you’:**
  - On other hand, you leave no trace if you pass your secret to somebody else
- If someone logged in using your user name and password:
  - Can you **prove your innocence**?
  - Can you **prove that you did not divulge** your password?

Approaches to User Authentication

Something you know

- **A password does not authenticate a person:**
  - Successful authentication only implies the user knows a particular secret
  - No way of telling the difference between the legitimate user and an intruder who has obtained that user's password
- And **recall all the challenges already discussed**:
  - Password interception, memorising and guessing
  - Caching, phishing, spoofing, key-loggers, social engineering

Approaches to User Authentication

Something you hold

- The user has to present a **physical token** to be **authenticated**
- Examples:
  - A physical key that opens the lock of a door
  - A card or an identity tag
- A **physical token can be lost** or stolen
  - Anybody in its possession has the same rights as the legitimate owner

Approaches to User Authentication

Something you hold

- To increase security, physical tokens are often used in **combination** with:
  - **Something you know** (e.g. a PIN): may not prevent a fraudster from obtaining the information necessary to impersonate a legitimate user.
  - **Something you are** (e.g. a photo): a fraudster impersonate the legitimate user based on the photo or on the description of physical characteristics.
- Notice that these are examples of multi-factor authentication.

Approaches to User Authentication

Who you are: biometric schemes

- Use unique physical characteristics (traits, features) of a person:
  - Face, fingerprints, iris patterns, hand geometry, voice, etc.
- Enrolment of a fingerprint:
  - Pattern of ridges serves as the unique characteristic
  - First, *samples* of the user’s fingerprint are collected
  - *Features* are extracted from the samples and stored as *reference templates*
  - Several templates may be recorded for greater accuracy

Approaches to User Authentication

Who you are: biometric schemes

- Recognition of a fingerprint:
  - When the user logs on, a new reading of the fingerprint is taken
  - A template is once again derived from the reading
- Matching two templates:
  - Reference templates will hardly ever match precisely this new template
  - The matching algorithm measures similarities between templates
  - The user is accepted if the similarity is above a predefined threshold

Approaches to User Authentication

Who you are: problems of biometric schemes

- Authentication using password:
  - Clear reject or accept at each attempt.
- Authentication using biometry:
  - Based on similarity (exact matches are not to be expected).
- **False positives**: accepting the wrong user is a security problem.
- **False negatives**: rejecting a legitimate user creates embarrassment, and is a usability and (potentially) an availability problem.

Approaches to User Authentication

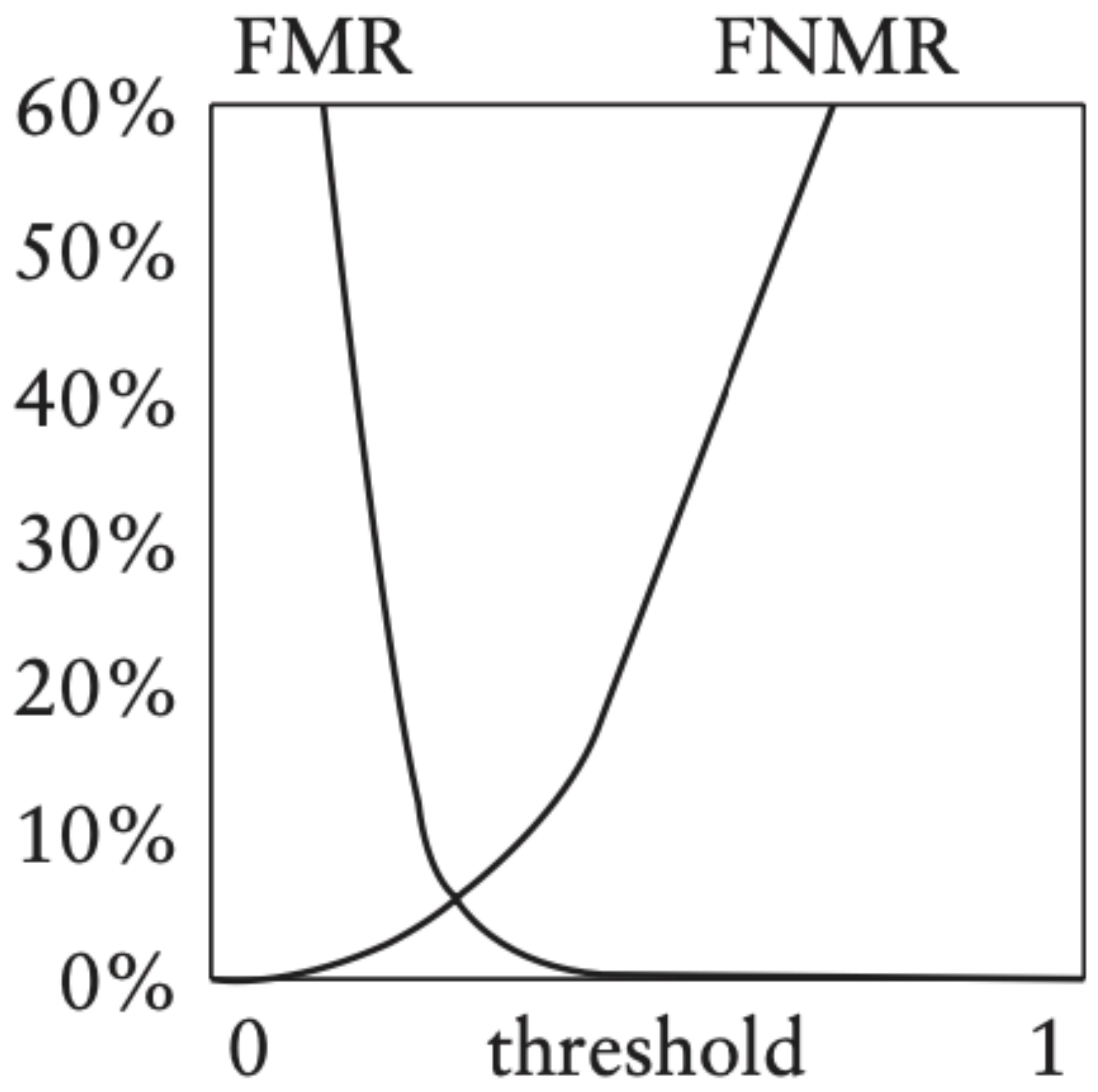
Who you are: error rates

- False Match Rate (FMR), and False Non-Match Rate (FNMR)

$$FMR = \frac{Number\ of\ Successful\ False\ Matches}{Number\ of\ Attempted\ False\ Matches}$$

$$FNMR = \frac{Number\ of\ Reject\ Geneuine\ Matches}{Number\ of\ Attempted\ Guenuine\ Matches}$$

- The lower the FMR, the higher the FNMR
- Equal Error Rate (EER): threshold where FMR = FNMR
- The lowest the EER, the better
- But acceptable threshold depends on the application



Approaches to User Authentication

Who you are: technologies and their EER

Technology	EER Range	Notes
Fingerprint (modern sensors)	~0.1% to 2%	Best systems can be very low, close to 0.1%
Face recognition (good systems)	~0.1% to 1%	With high-quality sensors, e.g., 3D face maps
Iris recognition	~0.01% to 0.1%	*One of the best in terms of low EER*
Voice recognition	~1% to 10%	Generally worse, because voices are variable
Vein (palm/hand vein scanners)	~0.01% to 0.1%	Extremely accurate; hard to fake
Behavioural biometrics (keystroke, gait)	~5% to 15%	Higher EER, much noisier signals

Approaches to User Authentication

Who you are: practical notes

- Lower EER often means:
  - Higher hardware costs (better sensors), and
  - More user inconvenience (longer scans, more precision needed).
- Security-sensitive environments (airports, military):
  - EER $< 0.1\%$.
- Consumer devices (phones, laptops):
  - Accept higher EERs for user convenience (~0.5%–2%).

Approaches to User Authentication

Who you are: other considerations

- Biometric traits may be unique but they are not secrets:
  - Fingerprints can often be retrieved from objects and then duplicated
  - Rubber fingers can defeat most fingerprint recognition systems
- Use of biometric authentication may be seen as unacceptable or undesirable

Approaches to User Authentication

What you do: behavioural security

- A user authenticates by performing a **mechanical task**:
    - The type of mechanical task must be **repeatable** and **specific** to an individual.
- Examples:
    - Keystrokes performed when handwriting a signature, pattern of walking (gait)
- Similar considerations to biometric schemes (can be seen as a case of “who you are”)
    - But behavioural traits are **more prone to forgery**
    - Hand-written signatures are relatively easy to forge by skilled criminals
    - For greater security, users could sign on devices that measure writing speed and pressure

Approaches to User Authentication

Where you are: location, location, location

- The system may also take into account **where you are**.
- Some operating systems only grant access only if a user logs in from a certain terminal:
    - An administrator may only be allowed to log in using the system's physical console
    - A regular user may only be allowed to log in using a particular workstation
- Decisions of this kind are **becoming more frequent** in mobile and distributed computing:
    - Remote logins may only be allowed from devices within office premisses
    - A precise GPS location may by required (using a trusted device) at the user end at login

User Authentication

Main approaches

CHARACTERISTIC	SOMETHING YOU KNOW	SOMETHING YOU HOLD	WHO YOU ARE
Based on	Shared secret	Physical possession	Intrinsic biometry
Assumptions	Cannot be revealed	Cannot be lost	Cannot be duplicated
Example	A password	A key	A fingerprint
Limitations	Becomes less secure with time; hard to remember	Can be lost; can be duplicated	Hard to replace; may be hard to accurately obtain
Combinations	Two-factor		
		Two-factor	
	Two-factor		🟦
	Three-factor		

User Authentication

Other approaches

CHARACTERISTIC	SOMETHING YOU DO	WHERE YOU ARE
Based on	Mechanical action	Physical or logical location
Assumptions	Repeatable but unique	Location be tracked
Example	Hand signature, sequence of key strokes, write on pressure-sensitive pad	An administrator may only be able login from a particular equipment
Limitations	May be forgeable	May be inaccurate or forgeable
Combinations	Can be use in multi-factor authentication	Can be use in multi-factor authentication

Approaches to User Authentication

Multi Factor Authentication

- E.g. A time-based one-time password (TOTP) application is relies on an initially shared seed (secret, something you know), stored in a user device (something you have), and on time as the moving factor.
- Side note: hash-based one-time password (HOTP) relies on an initially shared seed and on an event-based counter as the moving factor.
    - Often regarded as weaker than TOTP but stronger than passwords (susceptible to brute-force attacks).

Single Sign-On (SSO)

- **Multiple authentication** can be a significant **usability issue**:
  - Multiple passwords to memorise, and to repeatedly enter at login
- A **Single Sign-On (SSO)** service addresses this problem:
  - If not authenticated, the app redirects the user to a SSO server (authentication provider)
  - The user logs in (signs on) the SSO server (password, biometric, …)
  - If successful, the SSO server redirects the user back to the app, providing it a proof of authentication (token)
  - When accessing another app, this one just checks if the user possesses an authentication token (no need to re-enter credentials)

Single Sign-On

New security concernes

- SSO adds to your **convenience**
  - Ease of use is an important factor in making IT systems really useful
- Unfortunately, convenient often introduce **new security concerns**
  - How do you protect the SSO token?
  - If the SSO account is compromised all linked accounts are at risk
- Users and system designers have to **balance convenience and security**
  - Multi Factor Authentication (**MFA**) is often considered to be **crucial**

Single Sign-On

Common technologies

- SAML (older, but still widely used in enterprise)
- OAuth 2.0 (most modern web apps use this)
- OpenID Connect (OAuth 2.0 + identity layer; very popular now)
- Kerberos (used in internal corporate networks)

Protection Against Identity Theft

- **Monitor accounts**: Regularly review bank statements, credit card statements, and other financial accounts for any unauthorised transactions.
- **Change passwords**: Update passwords for all online accounts. Use strong, unique passwords and consider using a password manager to securely store them.
- **Enable Two-Factor Authentication**: Enable two-factor authentication whenever possible to add an extra layer of security to accounts.
- **Be cautious of phishing attempts**: Stay vigilant against phishing emails, calls, or texts that may try to trick you into revealing sensitive information. Be skeptical of any unsolicited communications and verify the source before sharing any personal data
- **Consider enrolling in an Identity Monitoring service**: These services help keep personal information safe, with early alerts if personal data is found online, and on the dark web.