Securing Devices

AP Cybersecurity ▪ Unit 4

AP Cybersecurity

What we will cover

1. Devices & malware types
2. Hashing & salt
3. Password attacks
4. Authentication factors
5. Protecting devices
6. Detecting attacks

1 Malware

Devices, and the malware that hunts them

A device is any computer – server, laptop, phone, or an **embedded computer** 嵌入式计算机. Everyday embedded devices are **IoT** 物联网: water pumps to washing machines.

> The main threat is **malware** 恶意软件 – and each type is defined by one trait.

Fingerprint auth

The malware types, by how each behaves

virus 病毒	attaches to a file and needs a human to run it
worm 蠕虫	spreads by itself, with no human action – which is why it spreads so fast
spyware 间谍软件	secretly tracks what you do
logic bomb 逻辑炸弹	lies dormant and triggers only when a condition is met – a date, a version, a name removed from payroll
rootkit 根工具包	buries itself in the operating system and can make itself invisible to the tools looking for it
ransomware 勒索软件	encrypts the victim's data and demands payment for the key

> Adversaries get in through **unpatched software** 未打补丁的软件, weak passwords, unprotected BIOS/UEFI settings and open ports – and **IoT** 物联网 devices, which are rarely patched at all.

The malware families worth naming

Trojan	A **trojan** arrives disguised as something wanted – an installer, a document – and runs with whatever rights the user had.
Remote access trojan	A **remote access trojan (RAT)** gives the attacker an interactive session: files, camera, keystrokes.
Ransomware	Encrypts the files and sells the key. Backups, offline and tested, are the only reliable answer.
Worm	Spreads by itself across a network with no user action – which is why patching is a defence against it and awareness is not.

> An **indicator of compromise (IOC)** is the evidence left behind: a hash, a domain, a registry key. Detection is built from them.

Fileless malware, and what adversaries exploit

Most malware is a file. **Fileless malware** 无文件恶意软件 is different: it lives only in **RAM** 内存 and abuses legitimate programs already on the device.

> It leaves no file for a scanner to find – which is exactly why signature scanning misses it.

Security key

2 Hashing

The cryptographic hash function

A **hash function** 密码散列函数 is one-way maths turning any input into a fixed-length **hash** 散列值. Three vital properties:

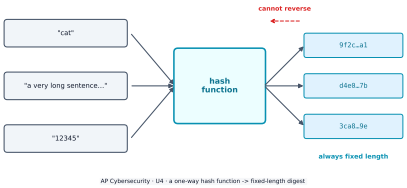
- **collision resistant** 抗碰撞 – hard to find two inputs with one output
- **pre-image resistance** 抗原像 – you cannot work backwards
- repeatable – the same input always gives the same hash

Real hash functions, and how they are attacked

SHA 安全散列算法	the Secure Hash Algorithm family – SHA-256 and SHA-512 are today's standard
brute force 暴力破解	tries every possible password in turn – guaranteed eventually, and the cost grows explosively with length
dictionary attack 字典攻击	tries common words and known passwords first, because most people choose guessable ones
rainbow table 彩虹表	precomputed hashes looked up instantly – which is precisely what a salt defeats
collision 碰撞	two inputs with the same hash; finding one breaks the function, which is why MD5 and SHA-1 are retired

> Length beats complexity against brute force, because each extra character multiplies the search space rather than adding to it.

The cryptographic hash function, in detail



fixed length, and no way back

> A **hash function** 密码散列函数 is one-way maths turning any input into a fixed-length **hash** 散列值.

Worked example – why salt matters

Two users both choose `sunshine`

✗ **No salt:** both stored hashes are identical – cracking one instantly cracks the other. ✓

With salt 盐值: give each a few random bits – x7 and q2. The service hashes `sunshinex7` and `sunshineq2`, so the stored hashes look completely different.

> The adversary must now attack each account separately – which is why a stolen hash database is far less dangerous when salted.

Never say a service "stores the password". It stores the salted hash.

Password attacks – and their log signatures

Password spraying 密码喷洒
one common password against many accounts
log: many users failing from one IP

Credential stuffing 撞库
stolen or default credentials
log: a burst of default logins

Rainbow table 彩虹表
a precomputed table of passwords and hashes

Guessing
log: one user + many wrong passwords

> Online attacks guess against a live login. Offline attacks steal the hash database and crack it on the adversary's own machine – so they cannot be detected. A favourite exam gotcha.

Each password-policy setting slows a specific attack

complexity 复杂度	a character from each set – upper, lower, digit, special. Slows brute force and dictionary attacks
minimum length 最小长度	*N* characters. Length matters more than anything else: brute-force cost grows exponentially with it
maximum age 最长有效期	a forced change every 90–120 days – limits how long a stolen password stays useful
password history 密码历史	stores the last 5–10 hashes and blocks reuse, stopping recycling of an old, possibly leaked, password
lockout 锁定	locks the account after 3–5 wrong tries – ends online guessing outright

> One subtlety worth a mark: some national standards now advise against forced expiry, because regular changes push users into predictable patterns like `Password1`, `Password2`.

Authentication factors

something you **know**
a password

something you **have**
a token or phone

something you **are**
a **biometric** 生物特征

somewhere you **are**
a location factor

> **MFA** 多因素身份验证 combines two or more categories – a fingerprint plus a password, not two passwords.

Authentication, and the devices to defend

Multifactor authentication	**Multifactor authentication (MFA)** combines factors: something you know, have, and are. A stolen password alone is then not enough.
Password manager	A **password manager** makes long unique passwords practical – which removes reuse, the single biggest cause of account takeover.
Endpoint detection	**Endpoint detection and response (EDR)** watches behaviour on the device itself and can isolate it from the network.
Internet of things	**Internet of things (IoT)** devices ship with default credentials, rarely update, and sit inside the network – the weakest link in most homes.

> Defence in depth: assume each control fails, and ask what the next one catches.

3 Protect & detect

Protecting devices

Managerial sets the rules: an **acceptable use policy** 可接受使用政策, a password policy (length, reuse), a software installation policy.

Anti-malware 反恶意软件 – matches signatures, quarantines

each **patch** 补丁 closes a known hole first

a **host-based firewall** 主机防火墙 guards one device

> Patching closes known holes before adversaries can use them – most breaches use a hole with a patch already available.

Detecting attacks – indicators of compromise

Logs reveal an **indicator of compromise** 入侵指标 – evidence an adversary got in:

- **host-based** – unexpected processes, changed settings
- **file-based** – a file whose hash matches known malware
- **behaviour-based** – many failed logins, unusual hours

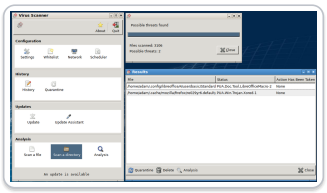
> Choosing a method weighs performance (signature is lighter – better for weak devices), cost (**EDR** 端点检测与响应 is powerful but expensive), and how sensitive the device is.

Detecting what got past the controls

indicator of compromise 入侵指标	IoC: evidence a system has already been breached – an unexpected outbound connection, a new admin account, a changed system file
EDR 端点检测与响应	endpoint detection and response: continuously watches each device's behaviour, and can isolate one automatically
keylogger 键盘记录器	records keystrokes – an IoC in itself, and a way credentials leave
logging	an IoC is only visible if something recorded it, which is why log retention is a control in its own right

> Prevention fails eventually. Detection is what turns a breach lasting months into one lasting hours, and that difference is most of the cost.

Protecting Devices, continued



> Anti-malware software scans files against a signature database and quarantines any matches – this scan has flagged two threats

Exam tips

- Know each **malware type by its defining trait**: a worm self-spreads, a virus needs a user, ransomware encrypts for money, a RAT gives remote control, a rootkit hides.
- A hash is **one-way and fixed-length**; **salt** makes identical passwords hash differently. Never say a service "stores the password" - it stores the **salted hash**.
- Name real algorithms: **SHA-256/SHA-512** are current; **MD5** and **SHA-1** are **deprecated** because efficient collision attacks exist.

Exam tips (continued)

- Match the password attack to its log signature: one user + many wrong passwords = guessing; many users + one IP = **spraying**; default credentials = **stuffing**.
- Sort authentication factors into **know / have / are / where**, and remember **MFA** combines two or more - a fingerprint plus a password, not two passwords.
- **Offline password attacks cannot be detected** because the cracking happens on the adversary's machine - a favourite exam "gotcha".

4 Recap

Unit 4 – key takeaways

1. **One trait each**
   worm self-spreads, virus needs a user, rootkit hides
2. **Salted hash**
   one-way, fixed length – never "stores the password"
3. **Log signatures**
   one user many passwords = guessing; many users one IP = spraying
4. **MFA**
   two different categories – not two passwords

Check yourself

1. Worm or virus: which needs a user to open a file?
2. Why can an offline password attack not be detected?
3. Are a password and a PIN multifactor authentication?
4. What does salt stop two identical passwords from doing?

Answers **1.** virus **2.** it runs on the adversary's own machine **3.** no – both are "something you know" **4.** producing identical hashes