



The Enigma Machine & Cryptanalysis

When math, machines, and mistakes changed the course of war

CR4CK1NG-THE-C0D3

Today's Roadmap

- The story behind Enigma
- Cryptanalysis of Enigma
- Manual vs Automated Cryptanalysis
- Hands-on Cryptanalysis using CrypTool



Video unavailable
[Watch on YouTube](#)



The Enigma

“I like solving problems, Commander. And Enigma is the most difficult problem in the world.” - Alan Turing

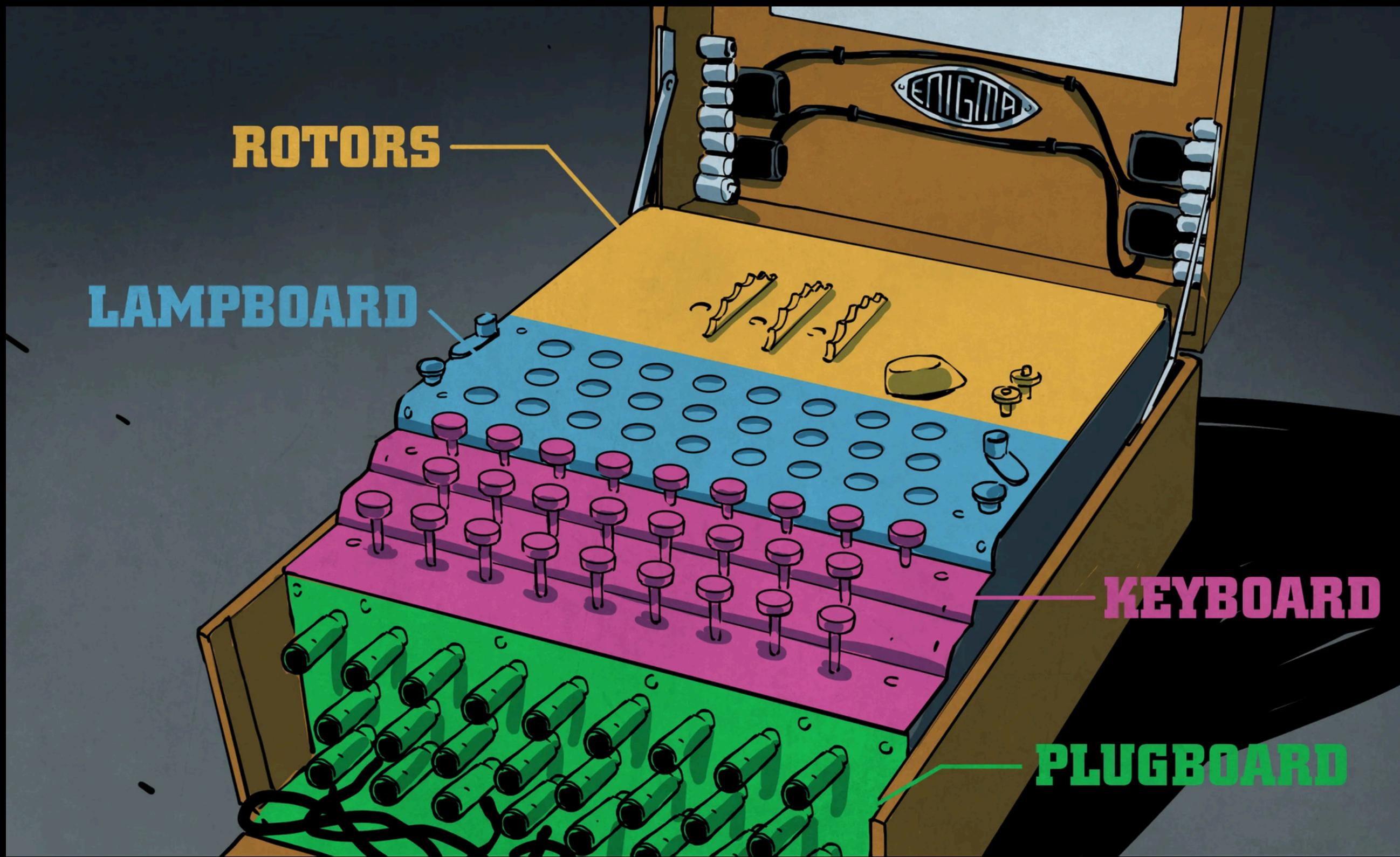
- Used by the Germans in WWII to encrypt military communication.
- Replaced simple substitution ciphers with rotating electrical rotors.
- Each key press changed the substitution pattern → no letter ever encrypted to itself!
- Believed to be unbreakable due to billions of combinations.



How did it work?

- 3 to 5 rotors, each rotating after every keystroke.
- A plugboard (Steckerbrett) swapped letters for added complexity.
- The machine was mechanical + electrical – both analog and digital thinking in one.

The machine



Lets look at its input and output

<https://matheusportela.com/enigma/>



The screenshot shows a dark-themed website for an Enigma simulator. At the top, there is a navigation bar with three links: "ABOUT", "BLOG", and "PORTFOLIO". To the right of the links are four social media icons: a blue square with a white feed symbol, a blue octagon with a white GitHub logo, a blue square with a white LinkedIn logo, and a blue square with a white envelope icon. Below the navigation bar, the main title "ENIGMA SIMULATOR" is displayed in large, bold, white capital letters. Underneath the title are two large, rectangular input fields: one labeled "INPUT" and one labeled "OUTPUT", both in white capital letters. At the bottom of the page is a "Show Settings" button in a white box with a black border. The footer contains the text "Developed by [Matheus V. Portela](#)" in a small, light gray font.

ABOUT BLOG PORTFOLIO

ENIGMA SIMULATOR

INPUT

OUTPUT

Show Settings

Developed by [Matheus V. Portela](#)

Why did it seem unbreakable?

- Over 150 quintillion possible key combinations (10^{23}).
- Daily key settings changed rotor order, ring settings, and plugboard pairs.
- Messages looked completely random – no visible patterns.
- Operators were trained to believe it was mathematically perfect.

Lets try our hands on it

<https://piotte13.github.io/enigma-cipher/>

PLUG SETTING: _____ [10 PAIRS ONLY] RESET MACHINE:

RING SETTING: << 0 >> << 0 >> << 0 >>

<< B >> << III >> << II >> << I >> ETW PlugBoard

The diagram illustrates the internal structure of an Enigma cipher machine. It features five rotors labeled A, B, II, III, and I from left to right. Each rotor has 26 contacts labeled with letters A through Z. The rotors are interconnected by a complex network of wires, with some contacts highlighted in red. Above the rotors, there are four rings labeled << B >>, << III >>, << II >>, and << I >>. Below the rotors, there is an External Wire Connector (ETW) and a PlugBoard. The ETW and PlugBoard also have 26 contacts each, labeled with letters A through Z. The entire assembly is contained within a dark green frame with various control buttons and indicators at the top and bottom.

This emulator uses a browser based Python interpreter (codeskulptor), and works with Firefox & Opera browsers.

LAMP BOARD

Q	W	E	R	T	Z	U	I	O
A	S	D	F	G	H	J	K	
P	Y	X	C	V	B	N	M	

KEY BOARD

Q	W	E	R	T	Z	U	I	O
A	S	D	F	G	H	J	K	
P	Y	X	C	V	B	N	M	

INPUTS:

OUTPUTS:

Manual Cryptanalysis

“Brains Before Machines”

- Relied on human logic, patterns, and language frequency.
- Cryptanalysts exploited operator habits and repeated phrases.
- “Heil Hitler” and weather reports gave starting points.
- Used cribs – guessed plaintext fragments to deduce wiring.



Automated Cryptanalysis

“When Logic Meets Machines”

- Alan Turing and his team at Bletchley Park built the Bombe.
- It systematically tested rotor positions using parallel circuits.
- Reduced days of manual work to hours.
- Introduced mechanical brute force + logic pruning
 - the basis for today's attacks.

Automated Cryptanalysis

Some extensive modern software tools:

The screenshot shows a GitHub repository page for 'Tools-for-Cryptanalysis'. The repository is public and has 13 watchers, 52 forks, and 215 stars. It contains 1 branch and 0 tags. The 'Code' tab is selected. The repository's history shows 59 commits from various contributors, including Deadlyelder, hadipourh, and others. The commits are dated between 6 years ago and 8 years ago and involve adding submodules and tools for cryptanalysis. The right sidebar provides information about the repository, including its purpose (providing tools for cryptography and cryptanalysis), tags (cryptography, crypto, cryptanalysis, cryptography-tools), and links to Readme, Unlicense license, Contributing, Activity, and reporting issues.

Deadlyelder / Tools-for-Cryptanalysis

Type ⌘ to search

Code Issues 1 Pull requests 1 Actions Projects Security Insights

Watch 13 Fork 52 Star 215

master 1 Branch 0 Tags Go to file Add file Code

Deadlyelder Merge pull request #3 from hadipourh/master 6529d80 · 6 years ago 59 Commits

AIPATH Added AIPATH 8 years ago

Ascon_test @ 285ba8e Add submodules 7 years ago

CRAFT-Integral-Distinguisher @ fff5e8b add craft-integral and mibs-integral submodules 6 years ago

CodingTool Added codingtool 8 years ago

MIBS-Integral-Cryptanalysis-Basd-on-Divisi... add craft-integral and mibs-integral submodules 6 years ago

MILP_conditional_cube_attack @ 2000fbc Add submodules 7 years ago

PRESENT Linear Hull Added Present-linear-hull, codes 8 years ago

S-Box MILP tool Added Present-linear-hull, codes 8 years ago

S-function toolkit Added s-function toolkit 8 years ago

YoyoTricksAES @ 8200808 Add submodules 7 years ago

cado-nfs @ 32b7241 Added CADO-NFS 8 years ago

cryptosmt @ 62ecf61 Add cryptosmt 8 years ago

grainofsalt @ f82b2a2 Added Grainofsalt, sage, symaes 8 years ago

isd @ db3bbe7 Added Present-linear-hull, codes 8 years ago

lextool @ 0dc488f Added Lex toolkit 8 years ago

sha1_gpu_nearcollisionattacks @ bcde2b7 Add submodules 7 years ago

About

A repository that aims to provide tools for cryptography and cryptanalysis

cryptography crypto cryptanalysis

cryptography-tools

Readme

Unlicense license

Contributing

Activity

215 stars

13 watching

52 forks

Report repository

Releases

No releases published

Packages

No packages published

Contributors 2

Deadlyelder Sankalp

hadipourh Hosein Hadipour

Layers of Security

- Many believe stacking ciphers = stronger security (e.g., Caesar + Vigenère + RSA).
- But combining without understanding dependencies can create new vulnerabilities.
- Enigma layered rotors, plugboard, and reflector – yet this layering created predictable symmetry.
- Example flaw: the reflector ensured no letter ever encrypted to itself – cryptanalysts used this clue.
- Modern parallel: “Encryption inside encryption” (e.g., HTTPS + VPN) can still fail if both rely on weak keys.

So lets look at the largest factor of error - Humans

Case Study – Human Error

- Cryptographers at Bletchley Park didn't defeat Enigma through math – they exploited human shortcuts.
- Common mistakes made by German operators:
- Reusing keys and messages (e.g., “AAA,” “HHH,” or date-based keys).
- Predictable openings like “WETTERBERICHT” (“Weather Report”).
- Sending test messages repeatedly.
- These patterns helped Alan Turing’s team align cribs with ciphertext segments.
- Moral: The system was secure; the humans weren’t.

Lets try the crib the team at Bletchley park used

Example of crib analysis

Imagine the Allied forces intercept a German Luftwaffe message sent from a known airbase at 6:00 a.m.. They assume the daily weather report is in the message and will likely contain the German phrase **WETTERBERICHT**. 

The intercepted ciphertext:

PJZ CQP EGLV WXG YJFG

The guessed plaintext (crib):

WETTERBERICHT

The codebreakers would test the crib against the ciphertext, shifting its position one character at a time. In most positions, there would be a "clash"—a letter in the crib would align with the same letter in the ciphertext. Because of Enigma's flaw, this combination of rotor and plugboard settings is impossible for that position. 

Crib P J Z C Q P E

Position

Possible Crib W E T T E R B

Clash (T=T) X

Possible Crib W E T T E R

Clash (E=E)

Possible Crib W E T T E

No Clash

In this simplified example, by checking against the "no-letter-encrypts-to-itself" rule, the codebreakers quickly eliminate almost all possible locations for the crib, leaving just a few possibilities for the machine's settings. ⓘ

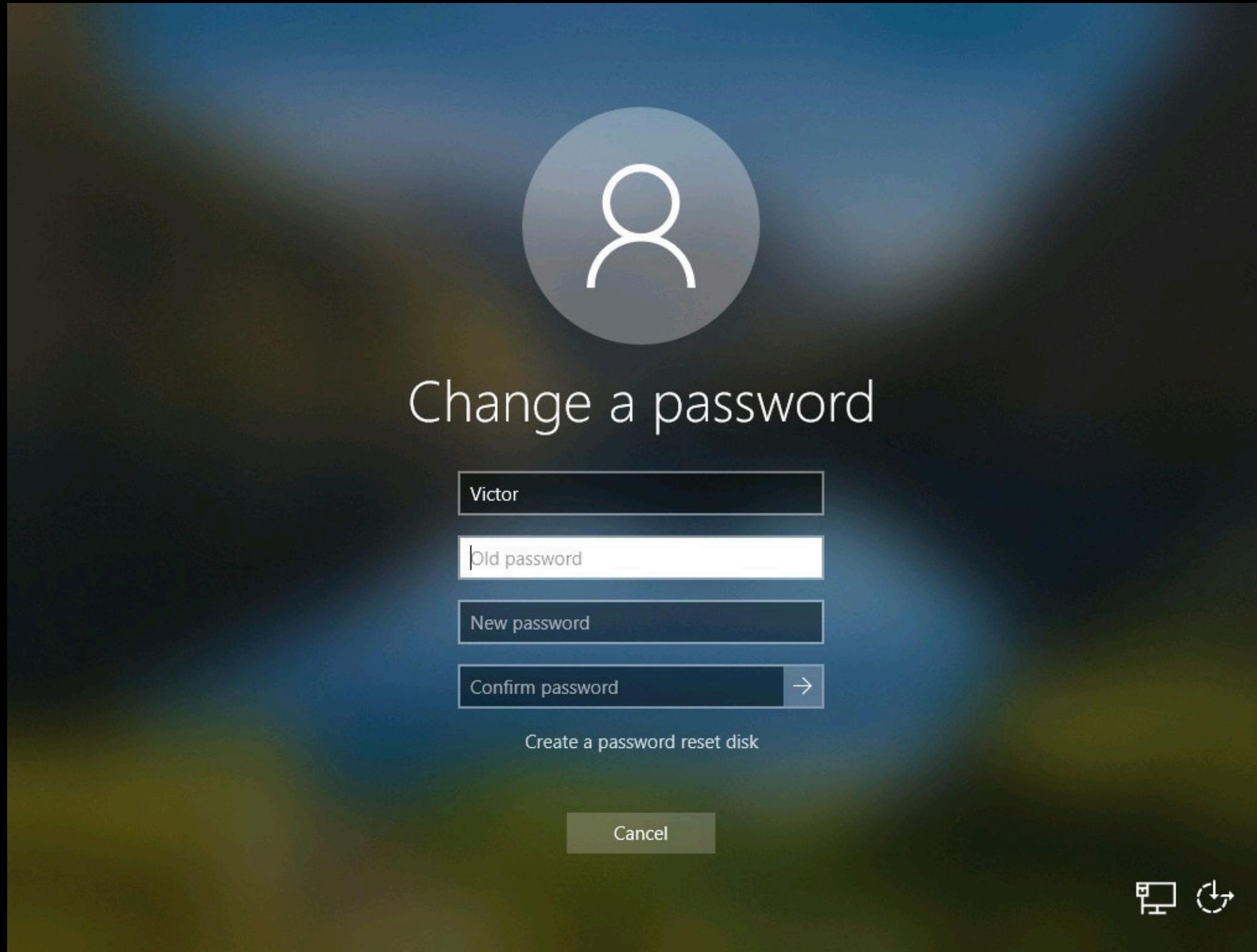
What lessons do we take back from this?

Operational Security (OpSec)

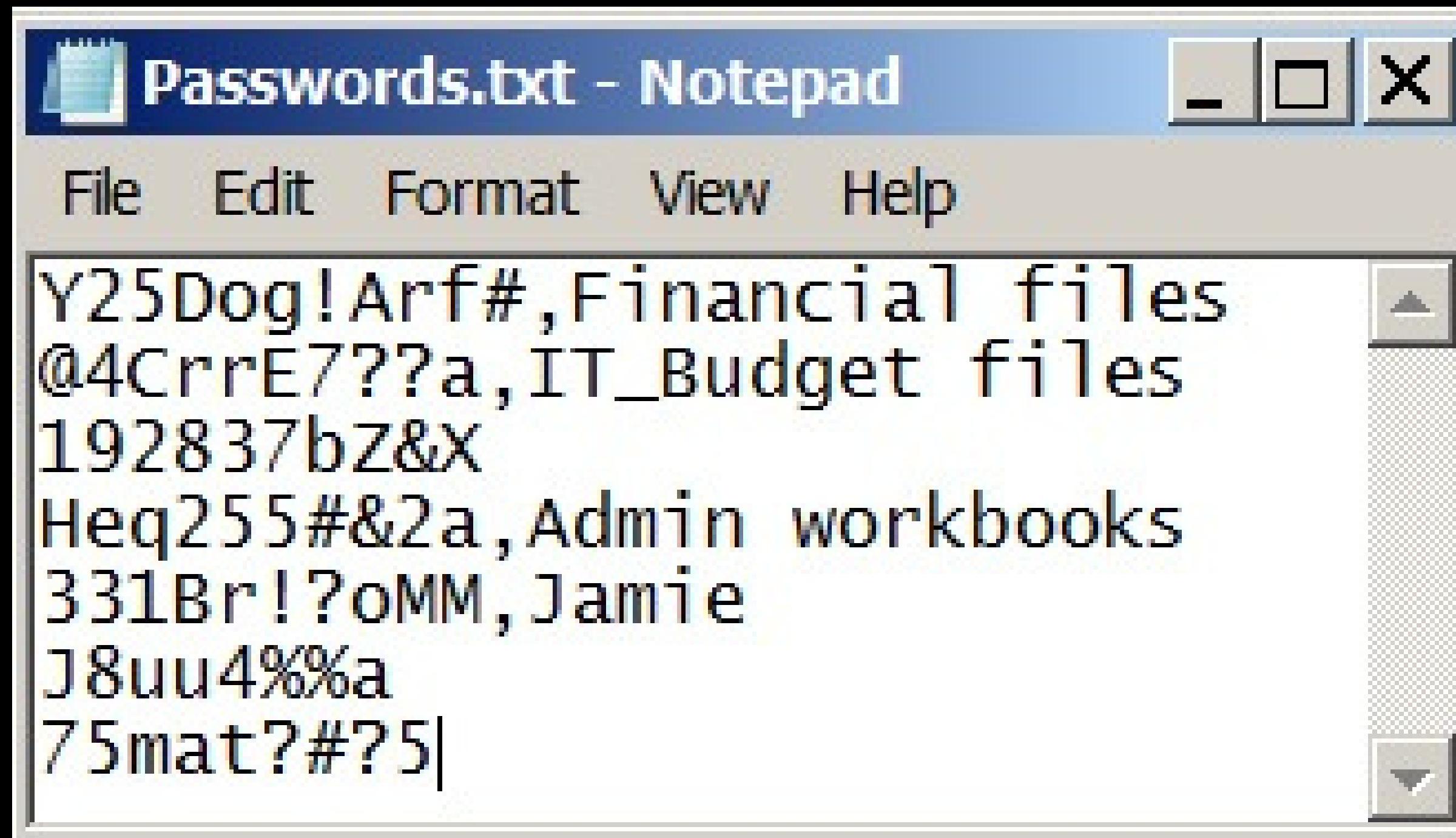
- OpSec = “Operational Security” = ensuring humans don’t compromise mathematical security.
- Key practices:
 - Rotation: Change keys often, use unique keys per session.
 - Separation: Keep encryption keys away from encrypted data.
 - Training: Teach users why “It’s just one password reuse” is dangerous.
- Poor OpSec turned Enigma’s strength into a weakness.
- Today’s crypto can fail even with AES-256 – if private keys are stored in plaintext.

Some good digital habits

Change your passwords often!!!



do not write your passwords in text files!



Use good password managers

The image displays three screenshots illustrating the use of a password manager:

- Left Screenshot:** A Mac OS X-style interface showing a sidebar with categories: All (1,167), Passkeys (3), Codes (10), Wi-Fi (98), Security (162), and Deleted (0). Below the sidebar, it shows "Shared Groups" with "Zap Testing" (1 item).
- Middle Screenshot:** A "Passkeys" screen titled "Passkeys" (3 Passkeys). It lists three items:
 - Passkeys**: harry@harryguinness.com (highlighted in blue)
 - Passkeys.io: harry@harryguinness.com
 - 1Password: harry@harryguinness.comA search bar is at the top.
- Right Screenshot:** A detailed view of a single passkey entry for "Passkeys".

Field	Value
Username	harry@harryguinness.com
Passkey	Created 24/07/2023
Website	passkeys.eu
Group	Not Shared

Security section:
A Passkey icon (a shield with a checkmark) is shown. The text explains: "A passkey is a secure way to sign in to your accounts using Touch ID. Passkeys provide more security than traditional passwords. [Learn more...](#)"

Real-World Parallels

- 2019 – Facebook: Stored hundreds of millions of user passwords in plaintext logs.
- 2021 – AWS S3 Bucket Leaks: Sensitive databases left publicly accessible.
- 2023 – IoT Devices: Hardcoded encryption keys discovered in firmware.
- All are modern echoes of Enigma: strong math, weak usage.
- Attackers don't need to crack AES if they can steal credentials or exploit human oversight.

Key Takeaways

- Even the most secure algorithm fails when humans are predictable.
- Both manual and automated cryptanalysis rely on pattern recognition.
- The future of security lies in minimizing human exposure to secret material.

Golden rule:

- *“Don’t trust users to be random. Design systems that don’t depend on them being perfect.”*

Wrap-Up Challenge – “Outthink the Machine”

- Imagine designing a cipher that anticipates human error.
- How will it handle reused keys?
- How will it resist predictable inputs?
- Create a mini concept (even pseudocode) for a “human-proof cipher.”
- Bonus points for creativity – “The Enigma 2.0” but smarter.
- *“A chain is only as strong as its weakest human.”*

Concept

- A “human-proof” cipher should not trust the user to:
 - remember or reuse secure keys,
 - avoid using the same password twice,
 - or keep track of nonces properly.
- So this cipher will generate all that automatically, and refuse to encrypt twice with the same key.

Hands on for implementing the same

Where can you try more of this out?



DE



CTO Applications

Knowledge ▾

Tools ▾

Source Code

Links

CrypTool-Online

Apps to explore, play around with, and learn about cryptology. For students, teachers, and anyone interested.

Search applications ▼

Historical encryption 11 apps



Atbash

Simple monoalphabetic substitution cipher originally used on the Hebrew alphabet



Caesar / ROT13

Famous shifting cipher used by Julius Caesar With Python code



Monoalphabetic Substitution

Cipher that replaces letters with letters/characters With Python code



Railfence / Redefence

Transposition cipher that uses a rail fence pattern

▼ Show 7 more

Modern encryption 4 apps

Display a menu



RSA (explained step by step)



OpenSSL