

# ENTROPY OF Argon2id combined with NIST primitives

## 1. English

ENTROPY OF Argon2id combined with NIST primitives

The Strength of the secure password generator by Argon2id in CryptoToolkit™

Apart from the Fact that it is BIP-39, This password generator possesses an expandable slider up to 128 characters (256 bits) with 4 criteria (Alpha, num, Upper/lower, Special characters, all of it with an expandable vault of 50 storage slots for PassPhrase or password.

The advantages of such an application within the application facing the vulnerability of current passwords which are breakable in a snap of a finger by powerful graphics cards.

- As seen previously, current AIs break simple passwords in a few minutes or even seconds
- The use of Argon2id (the best password hashing algorithm in the world) forces the hacker's computer or AI to consume a huge amount of memory and time to test a single passphrase. This makes "brute force" type attacks mathematically and financially impossible..

demonstration

The password generator of a tool like CryptoToolkit™ combines maximum entropy (via character generation) and extreme resistance to attacks (via Argon2id hashing).  
The strength of this system facing current vulnerabilities.



Why current passwords "melt" facing AIs

Phenomenal calculation speed: Modern graphics cards (GPUs) and optimized AIs test billions of combinations per second on classic hashes (like MD5 or SHA-256).

Human predictability: Humans create predictable variations (e.g.: Motdepasse123!). AIs trained on billions of data leaks guess these schemes in a few minutes.



Brute force neutralized: The slider effect (Up to 128 characters)

Gigantic entropy: A password of 128 characters combining the 4 criteria (Uppercase, lowercase, numbers, special characters) draws from an alphabet of about 90 to 95 characters.

Mathematically unbreakable: This represents  $95^{128}$  possible combinations (well beyond 256 bits of entropy).

Universal calculation time: Even by aligning all supercomputers and AIs on Earth, breaking such a key by brute force would take billions of years.



The ultimate shield: Argon2id

If this generator uses Argon2id to derive or secure keys/passphrases of its vault, the paradigm changes completely for the hacker:

Memory bottleneck (Memory-Hard): Unlike older algorithms, Argon2id demands a massive amount of RAM memory to validate a single attempt.

Anti-GPU and Anti-ASIC: Hackers use thousands of chips in parallel (GPU/ASIC) to go fast. Argon2id blocks this, because these chips have many calculation cores, but very little dedicated memory per core.

Resistance to side-channel attacks: The "id" variant combines resistance to timing attacks (Argon2i) and GPU attacks (Argon2d).

Financial asphyxiation: Testing a guess no longer takes a microsecond, but for example 0.5 seconds and requires 1 GB of RAM.

For a hacker, attempting to crack an Argon2id passphrase becomes financially and materially impossible (astronomical electricity and server bill).

☐ The advantages of the "Application within the application" integration

Having this generator and this vault of 50 slots directly integrated into CryptoToolkit™ offers major benefits:

Zero clipboard leak: No need to copy-paste your ultra-long passwords from a third-party website or an unsecured notepad, avoiding "keylogger" or "clipboard hijacker" viruses.

Data sovereignty: The vault of 50 slots is local, encrypted by Argon2id, and does not depend on a cloud server vulnerable to mass hacks (like historical leaks of other online managers).

BIP-39 / Passphrase complementarity: You can generate a master encryption key via the crypto security standard (BIP-39), then use the Argon2id manager to store complex "Passphrases" (the 13th or 25th word) protecting your crypto wallets against physical theft.

let us calculate together the theoretical time necessary for a mainstream AI to be able to brute-force one of these 128-character passwords:

Here is the entropy and theoretical time calculation, followed by configuration recommendations for Argon2id.



#### Part 1: Calculation of theoretical brute force time

For a password of 128 characters using the 4 criteria, the available alphabet (\$N\$) is about 95 characters (26 uppercase + 26 lowercase + 10 digits + 33 special characters).

##### 1. The number of combinations:

The total number of possible combinations is:

$$95^{128} \approx 4.37 \times 10^{253}$$

$95^{128}$  is written in full:

14 080 610 235 352 233 733 974 796 807 107 488 225 431 294 946 326 679 091 971 204 643  
836 621 273 151 625 656 703 817 627 574 060 339 920 877 177 547 088 562 233 219 059 963  
291 317 578 633 826 442 480 919 226 256 285 964 090 078 137 242 215 787 111 679 714 448  
326 688 998 187 048 752 763 761 427 786 448 621 191 084 384 918 212 890 625

That is a 254-digit number, consistent with the order of magnitude  $4.37 \times 10^{253}$  displayed on the screenshot.

In entropy bits, this represents  $\log_2(95^{128}) \approx 841$  bits of entropy (which far exceeds the threshold of 256 bits mentioned).

##### 1. The speed of a "mainstream AI" / Supercomputer

Even if a mainstream AI is fast at guessing human passwords, on a completely random password, it must test combinations one by one.

Suppose a hacker uses an excellent mainstream graphics card (RTX 4090 type) optimized to test 100 billion combinations per second ( $10^{11}$  attempts/sec)

Let us be even more extreme: imagine he has a supercomputer capable of testing 1000 peta-attempts per second ( $10^{18}$  attempts/sec)

##### 1. The result of the calculation time

By dividing the number of combinations by the speed of the supercomputer:

Here is the formula in text:

$$4.37 \times 10^{253} \div 10^{18} \text{ attempts/sec} = 4.37 \times 10^{235} \text{ seconds}$$

presentation as a fraction (numerator/denominator) rather than with the  $\div$  sign, in simple text it gives:

$$\frac{4.37 \times 10^{253}}{10^{18} \text{ attempts/sec}} = 4.37 \times 10^{235} \text{ seconds}$$

To convert this into years:  $4.37 \times 10^{235} \div (365.25 \times 24 \times 3600) \approx 1.38 \times 10^{228} \text{ years}$

Verdict: The age of the Universe is about  $1.38 \times 10^{10} \text{ years}$ .

**It would therefore take a time representing billions of billions of times the age of the Universe to break this password by brute force, making the attack mathematically obsolete....**