Understanding how to prevent

Sensitive Data Exposure

Dr Simon Greatrix

Just Trust The Internet!

- Lots of free advice
- Opinions to suit all tastes
- Also has pictures of cats!
- Not responsible for the collapse of civilization (so far)
- Google has excellent branding

Top tips for a successful presentation! (according to the internet)

- Start with a joke
- Imagine the audience naked
- Avoid showing emotion
- Use images to jazz things up

Obligatory famous historical data breaches

- 200,000 BCE to 6000 BCE : Tribal life with no privacy at all
- 600 BCE to 400CE: Ancient City culture view privacy as a bad thing
- 75 CE to 800 CE: There is no word for “privacy” in classical nor medieval Latin
- 1215 CE: Fourth Council Of Lateran makes confession mandatory
- 1450 CE: First use of “privacy” in English.
- 1700 CE: Solo beds
- 1890 CE: First use of “Right To Privacy”

“When people conceal their ways from one another … there no one will ever rightly gain neither their due honour nor office nor the justice that is befitting”

Socrates (470 to 399 BCE)

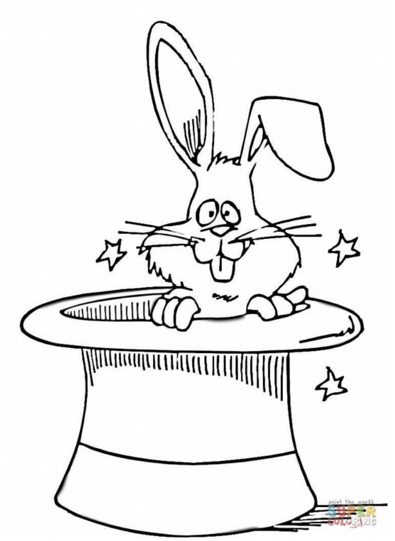
What are we actually expected to do?

- The Standards
  - OWASP
  - PCI-DSS

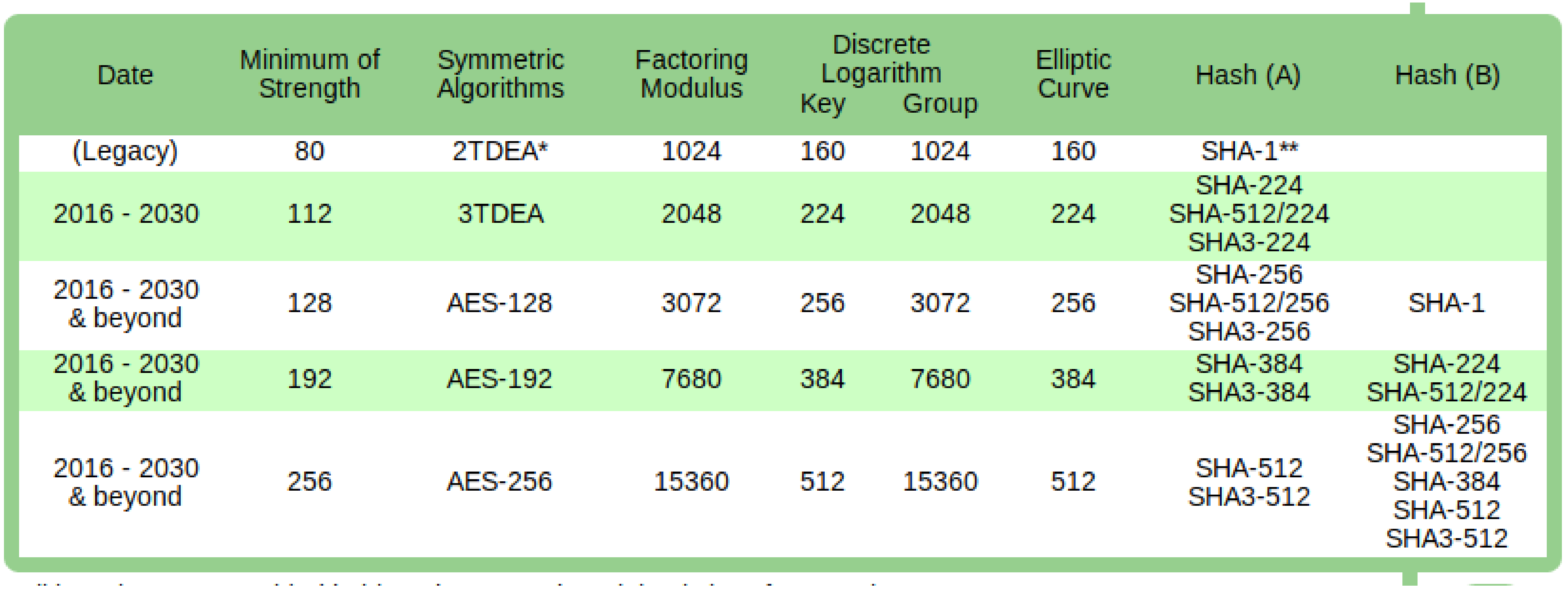
OWASP
The Open Web Application Security Project

PCI DSS COMPLIANT

- The Tools
  - Ciphers
  - Hashes



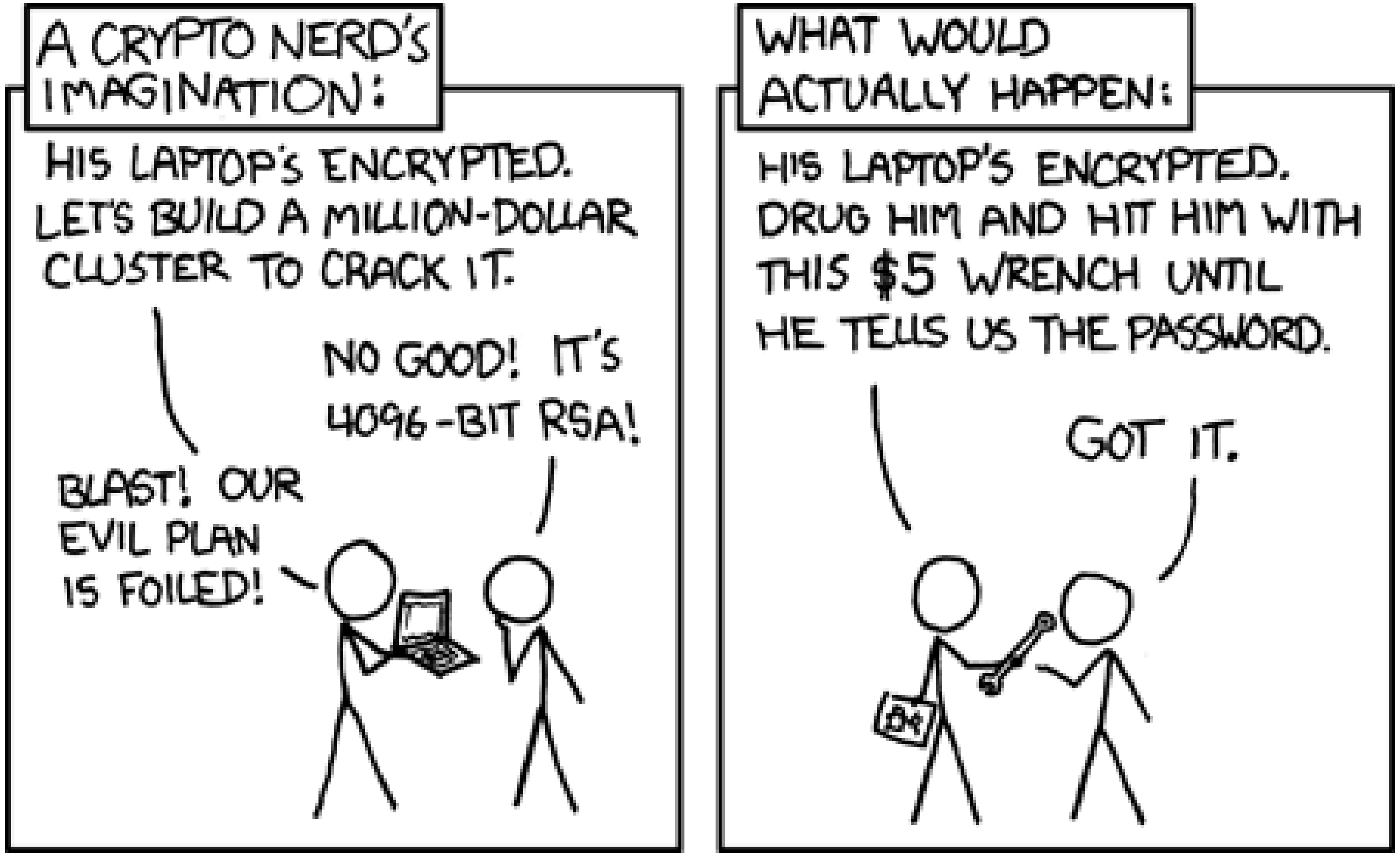
An aside: What is “strong encryption”?



Date	Minimum of Strength	Symmetric Algorithms	Factoring Modulus	Discrete Logarithm		Elliptic Curve	Hash (A)	Hash (B)
				Key	Group			
(Legacy)	80	2TDEA*	1024	160	1024	160	SHA-1**	
2016 - 2030	112	3TDEA	2048	224	2048	224	SHA-224 SHA-512/224 SHA3-224	
2016 - 2030 & beyond	128	AES-128	3072	256	3072	256	SHA-256 SHA-512/256 SHA3-256	SHA-1
2016 - 2030 & beyond	192	AES-192	7680	384	7680	384	SHA-384 SHA3-384	SHA-224 SHA-512/224
2016 - 2030 & beyond	256	AES-256	15360	512	15360	512	SHA-512 SHA3-512	SHA-256 SHA-512/256 SHA-384 SHA-512 SHA3-512

Source: https://www.keylength.com

Is this really a good idea?


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

https://xkcd.com/538/

Your Personal Information

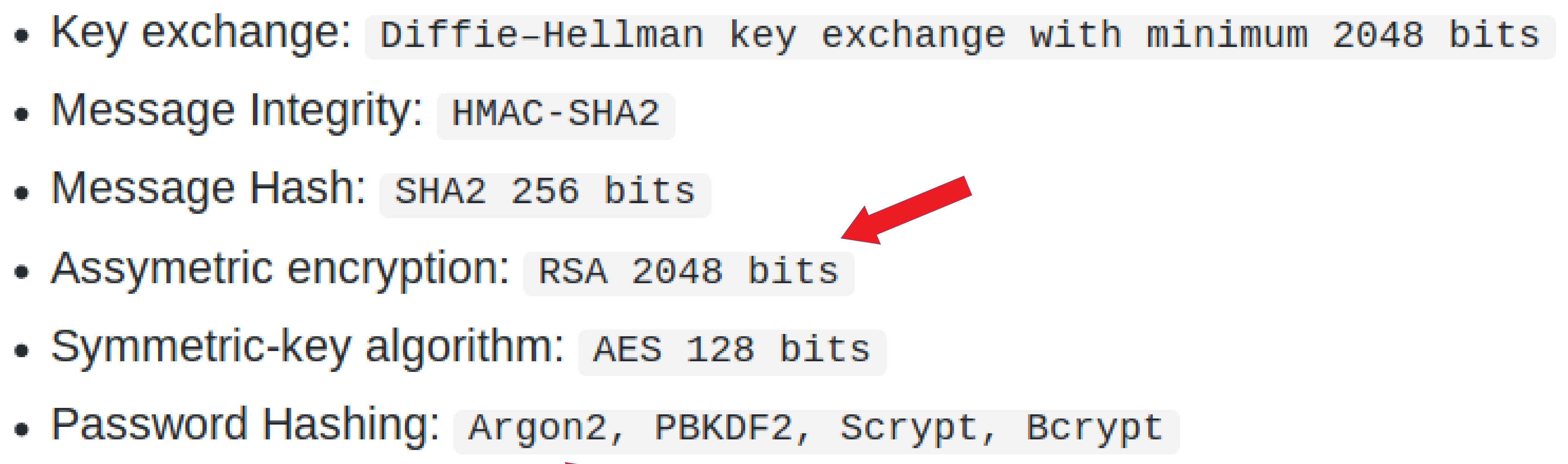
Please don’t send us your personal information. We do not want your personal information. We have a hard enough time keeping track of our own personal information, let alone yours.

If you tell us your name, or any identifying information, we will forget it immediately. The next time we see you, we’ll struggle to remember who you are, and try desperately to get through the conversation so we can go online and hopefully figure it out.

https://xkcd.com/1998/

OWASP’s Cryptographic Storage Cheat Sheet

- 17 Rules of which 9 have flaws
- Defines good encryption as:

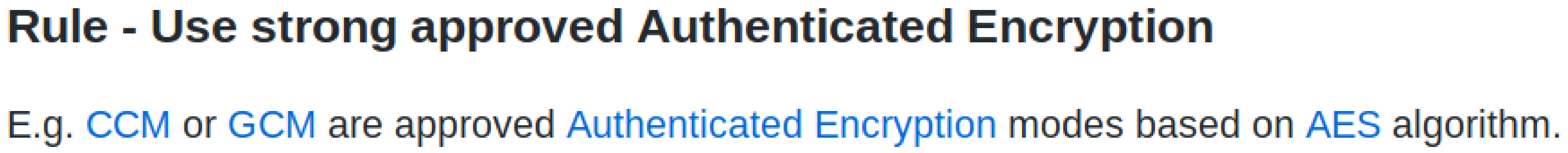


- Key exchange: `Diffie–Hellman key exchange with minimum 2048 bits`
- Message Integrity: `HMAC-SHA2`
- Message Hash: `SHA2 256 bits`
- Assymetric encryption: `RSA 2048 bits`
- Symmetric-key algorithm: `AES 128 bits`
- Password Hashing: `Argon2, PBKDF2, Scrypt, Bcrypt`

Another aside: What is Argon2?

- Winner of the Password Hashing Competition 2013
- The competition was organised by and judged by “some guys”
- ASIC resistance by being memory hard
  - ASICs are available up to 128MB. Authors recommend using 1GB
  - GPU resistance is unknown if memory is < 512kB
- Uses iterations to become time hard as well.
- Can be configured to be memory easy and time easy
  - Then there is no point using it
- Login is now a denial of service, or insecure

Issue 1: It is not “based on”



Rule - Use strong approved Authenticated Encryption

E.g. CCM or GCM are approved Authenticated Encryption modes based on AES algorithm.

My recommendation : AES with CFB, Encrypt-then-MAC, Hmac-SHA-256

Why AES?

DES

SEAL

HC-256

Rabbit

RC2

IDEA

RC4

Salsa

Nimbus

RC5

ICE

Blowfish

Scream

Hasty Pudding

AES

RC6

Threefish

Twofish

MARS

TEA

Crab

Lucifer

GOST

A5/1

CAST-128

Cobra

Hierocrypt

Zodiac

Camelia

XTEA

SNOW

Serpent

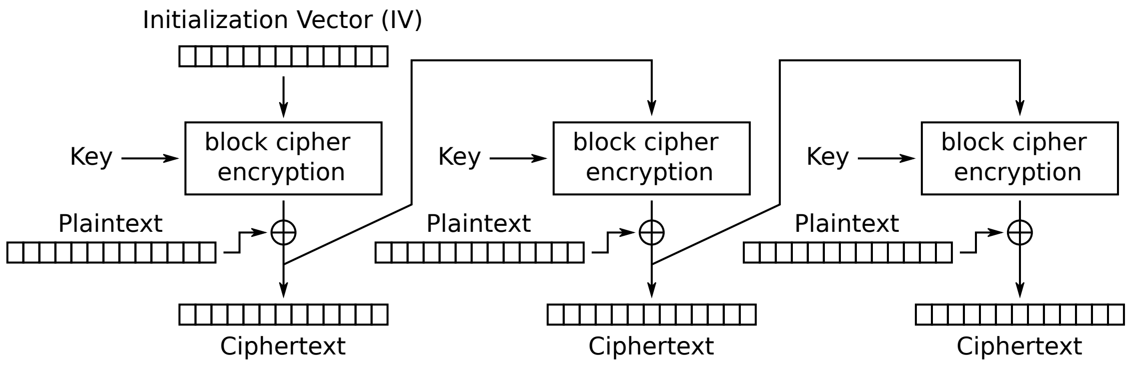
Grand Cru

ChaCha

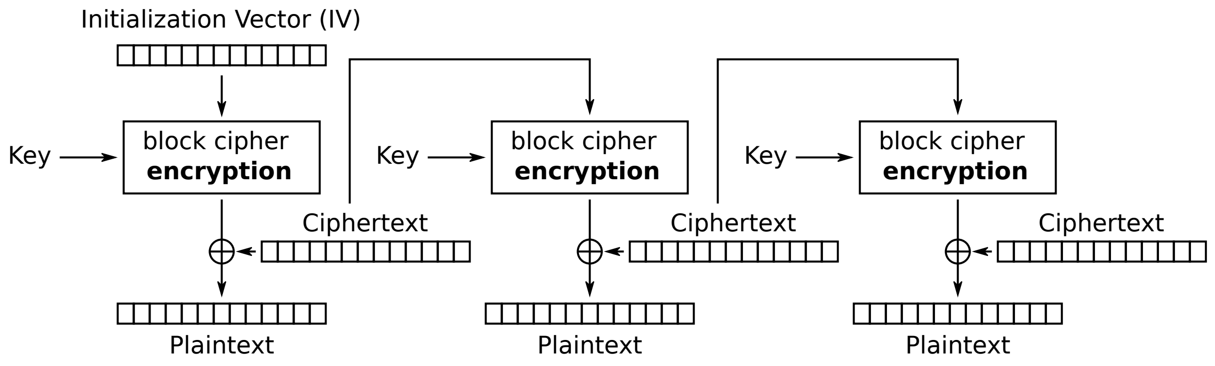
Why CFB?

- Electronic Code Book – reveals data structure
- Cipher Block Chaining – vulnerable to padding oracles
- Propagating Cipher Block Chaining – not widely studied
- Counter – Never repeat an IV. Key wears out sooner.
- Output Feedback – Individual bits can be flipped. OFBx is weak.
- Cipher Feedback – NIST + FIPS approved. No known vulnerabilities

What is CFB?

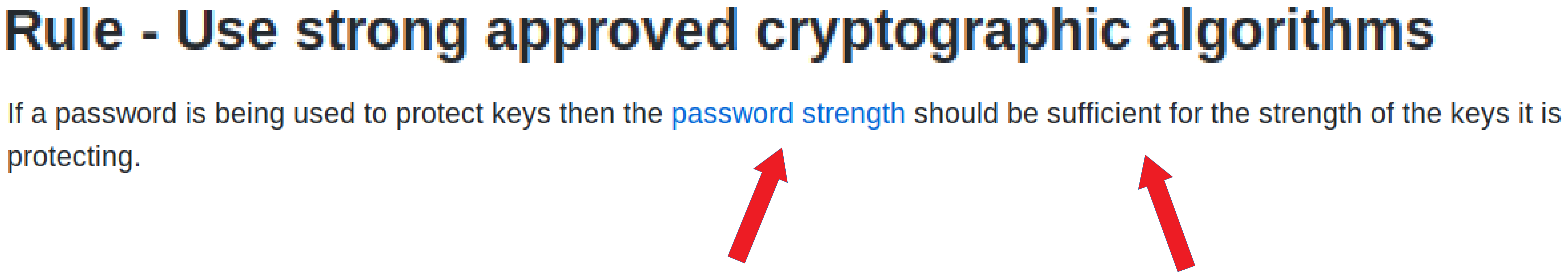


Cipher Feedback (CFB) mode encryption


Initialization Vector (IV)
Key
block cipher **encryption**
Ciphertext
Plaintext
Key
block cipher **encryption**
Ciphertext
Plaintext
Key
block cipher **encryption**
Ciphertext
Plaintext

Cipher Feedback (CFB) mode decryption

Issue 2: Password strength?

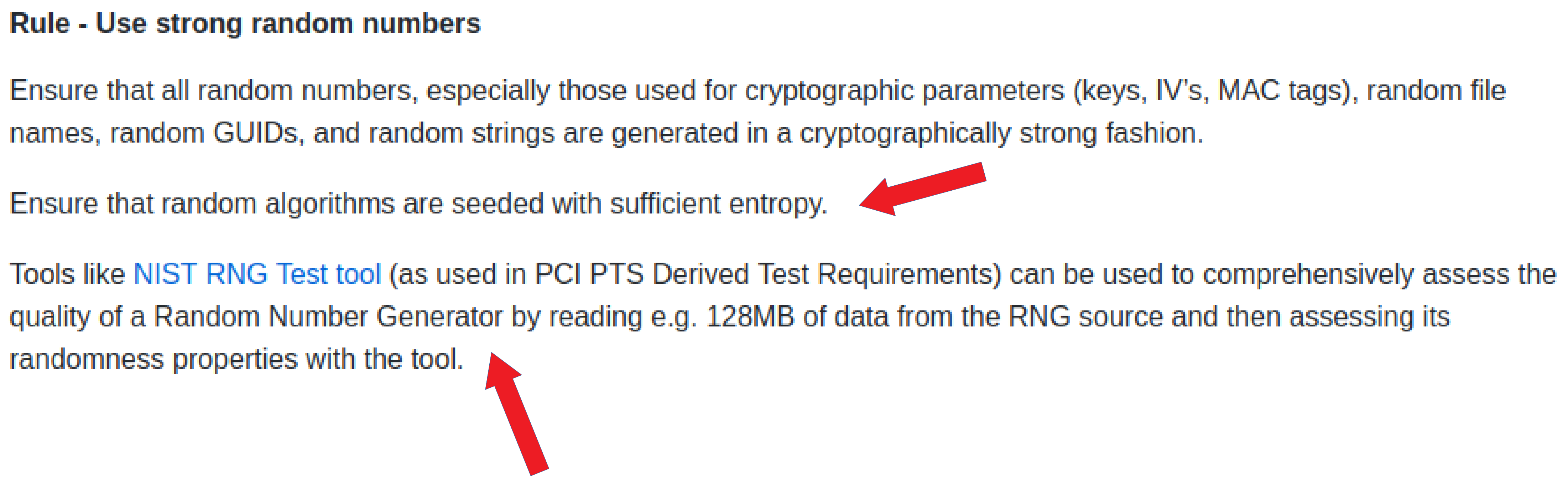


Rule - Use strong approved cryptographic algorithms

If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting.

- What is “sufficient”?
- 128 bit key requires a 20 character ASCII password
- 256 bit key requires a 39 character ASCII password
- This is the correct place to use Argon2!

Issue 3: Random numbers



Rule - Use strong random numbers

Ensure that all random numbers, especially those used for cryptographic parameters (keys, IV's, MAC tags), random file names, random GUIDs, and random strings are generated in a cryptographically strong fashion.

Ensure that random algorithms are seeded with sufficient entropy.

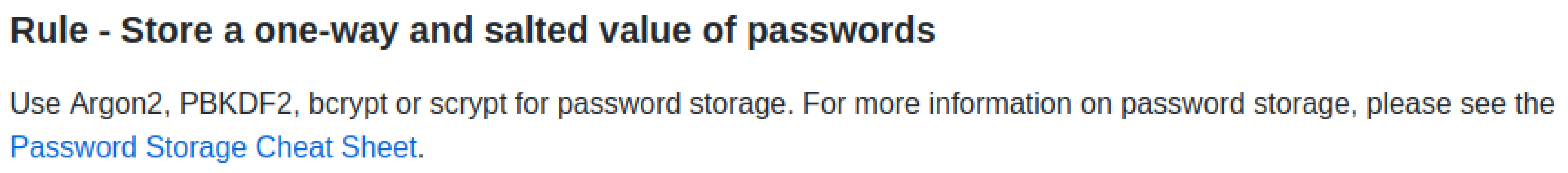
Tools like NIST RNG Test tool (as used in PCI PTS Derived Test Requirements) can be used to comprehensively assess the quality of a Random Number Generator by reading e.g. 128MB of data from the RNG source and then assessing its randomness properties with the tool.

- What is entropy?
- What is “sufficient entropy”?
- There is no such thing as a test for randomness

Issue 4 : Use Authenticated Encryption

- CCM or GCM?
- CCM : Counter with CBC Mac
  - Slow – have to encrypt twice
  - Can only be used on known length messages
  - First block vulnerable to IV changes
- GCM : Galois Counter Mode
  - Unknown number of weak keys
  - Best practice says change key if someone starts guessing MACs.
- Only 128 bits of authentication
- Should do what we said we'd do!
  (AES with CFB, Encrypt-then-MAC, Hmac-SHA-512/256)

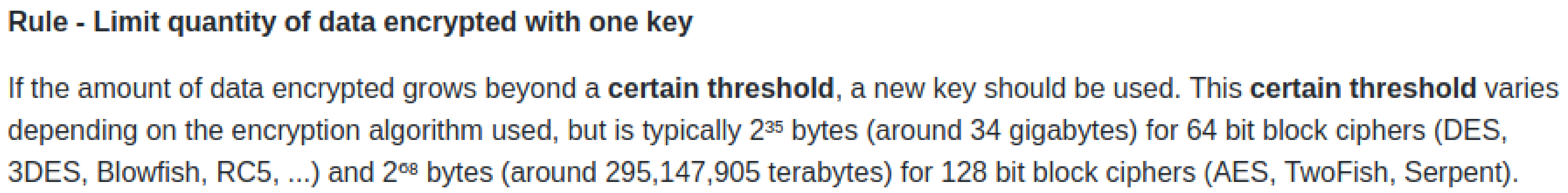
Issue 5: Password hashing


Rule - Store a one-way and salted value of passwords

Use Argon2, PBKDF2, bcrypt or scrypt for password storage. For more information on password storage, please see the Password Storage Cheat Sheet.

- Only one of these is even close to a good idea – we’ll get to that later.

Issue 6: How much data is safe?

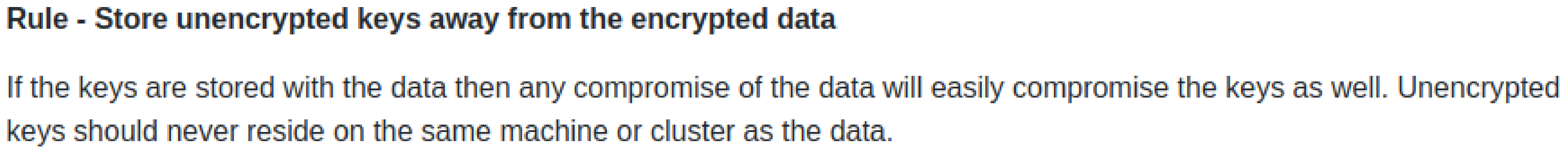


Rule - Limit quantity of data encrypted with one key

If the amount of data encrypted grows beyond a **certain threshold**, a new key should be used. This **certain threshold** varies depending on the encryption algorithm used, but is typically 2^{35} bytes (around 34 gigabytes) for 64 bit block ciphers (DES, 3DES, Blowfish, RC5, ...) and 2^{68} bytes (around 295,147,905 terabytes) for 128 bit block ciphers (AES, TwoFish, Serpent).

- The mode limits the amount of data too
- GCM
  - No more than 2^48 messages.
  - No more than 2^34 bytes per message.

Issue 7 : Practical?

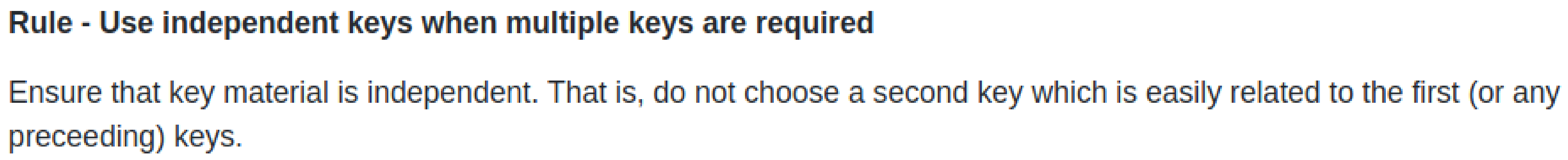


Rule - Store unencrypted keys away from the encrypted data

If the keys are stored with the data then any compromise of the data will easily compromise the keys as well. Unencrypted keys should never reside on the same machine or cluster as the data.

- If the key cannot be accessed, how is the key used?
- If the key can be accessed, how is it protected?
- There is no such thing as a cloud based key ceremony.

Issue 8

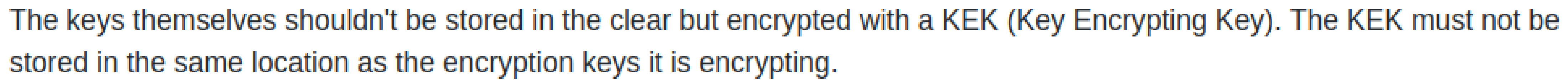


Rule - Use independent keys when multiple keys are required

Ensure that key material is independent. That is, do not choose a second key which is easily related to the first (or any preceeding) keys.

- Why not use a key derivation function?
- What is “independent”?
- Never “choose” a key

Issue 9: Key Encrypting Keys


The keys themselves shouldn't be stored in the clear but encrypted with a KEK (Key Encrypting Key). The KEK must not be stored in the same location as the encryption keys it is encrypting.

- How do you access the key encrypting key?
- Really should mention key wrapping.

My Recommendations

- Key exchange: Elliptic Curve Diffie Helman, followed by HKDF
- Message integrity: HMAC-SHA-512/256
- Message hash: SHA-512/256
- Asymmetric encryption: ECIES
- Symmetric encryption: AES with 128 bits in CFB mode
- Password Hashing: PBKDF2 with HMAC-SHA-256

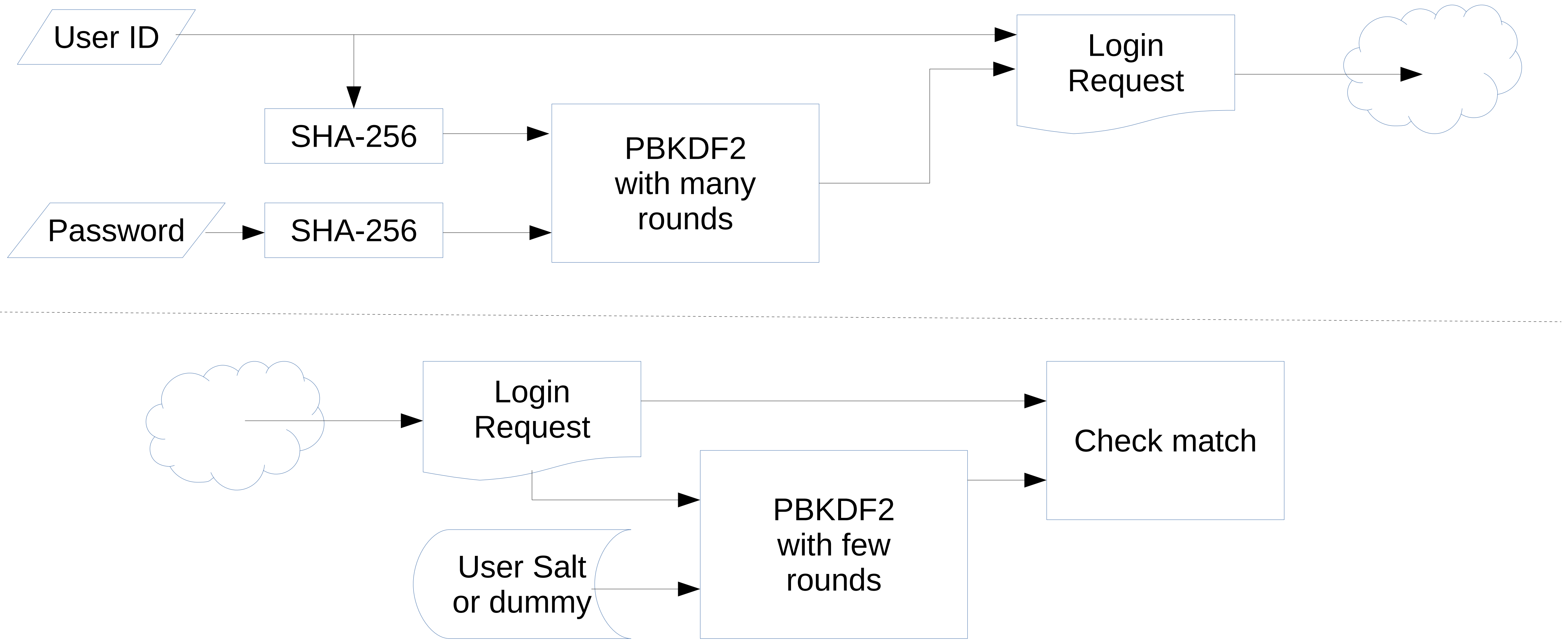
Password Storage 1

- Humans are rubbish at passwords
- Never store the password
- The algorithms:
  - Argon2
  - PBKDF2
  - Scrypt
  - bcrypt

Password Storage 2

- The OWASP cheat sheet is pretty good
- But is written by someone who loves Argon2
- PBKDF2 is required for FIPS certification
- PBKDF2 is approved by NIST
- Argon2 is approved by “some guys”
- With secure password storage, login is a denial of service
- How would I do it?

Password Storage



And finally...