Securely backing up GPG private keys ... to the cloud?

Joey Hess

Linux.Conf.Au 2017

Imagine if everyone used GPG

In a world where everyone has a GPG key...

In a world where everyone has a GPG key...

Everyone has a key backup problem.

GPG key backup methods

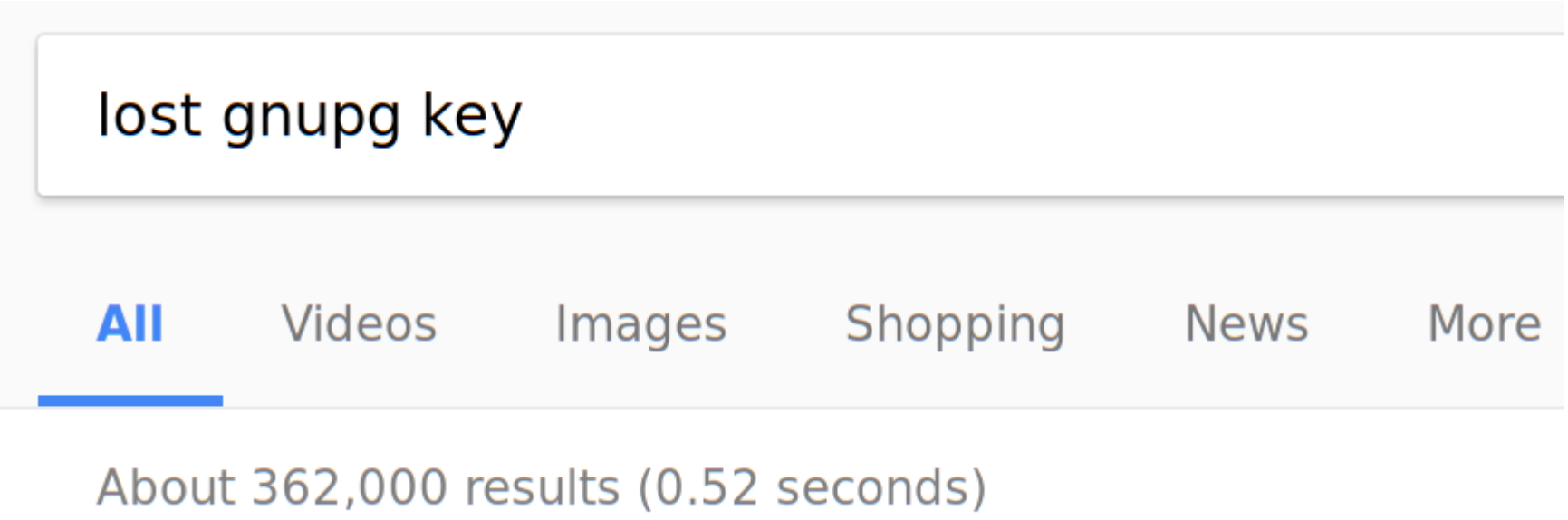
- Print out GPG key
  - `paperkey(1)`
  - Hard to back up
  - Hard to restore
- Backup $HOME to cloud storage
  - Not exactly secure
- Backup $HOME to encrypted cloud storage
  - `obnam(1) / attic(1)`
  - Encrypted using what key?
- Shard and store on USB drives, etc, scattered here and there
  - Not automated

GPG key backup methods

- Don’t back up GPG key
  - Common approach

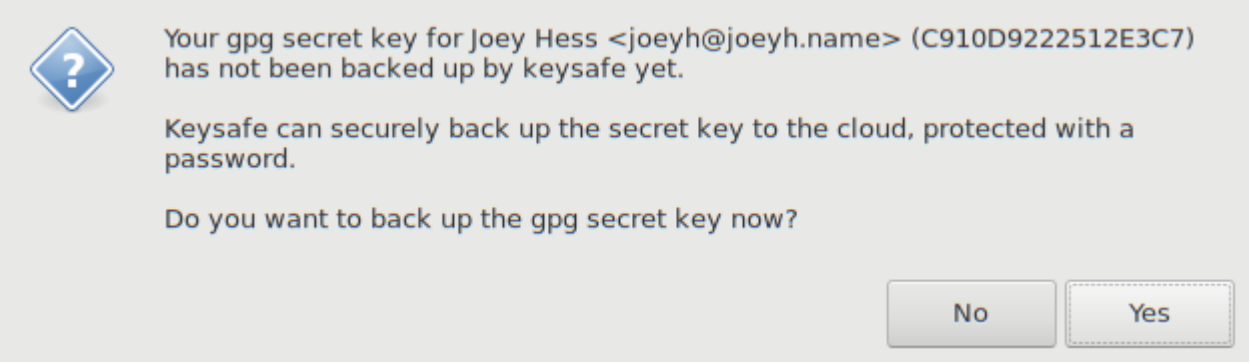
GPG key backup methods

- Don’t back up GPG key
  - Common approach

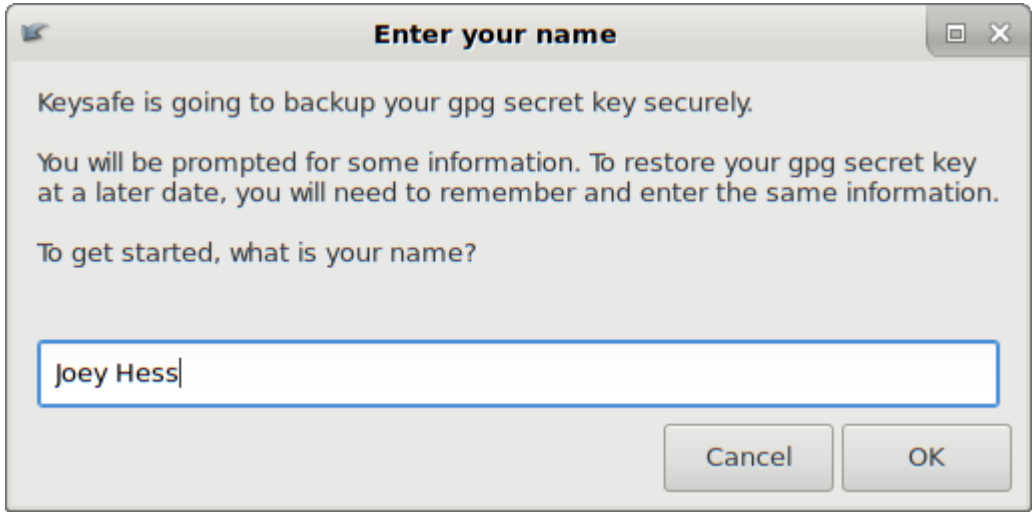


keysafe

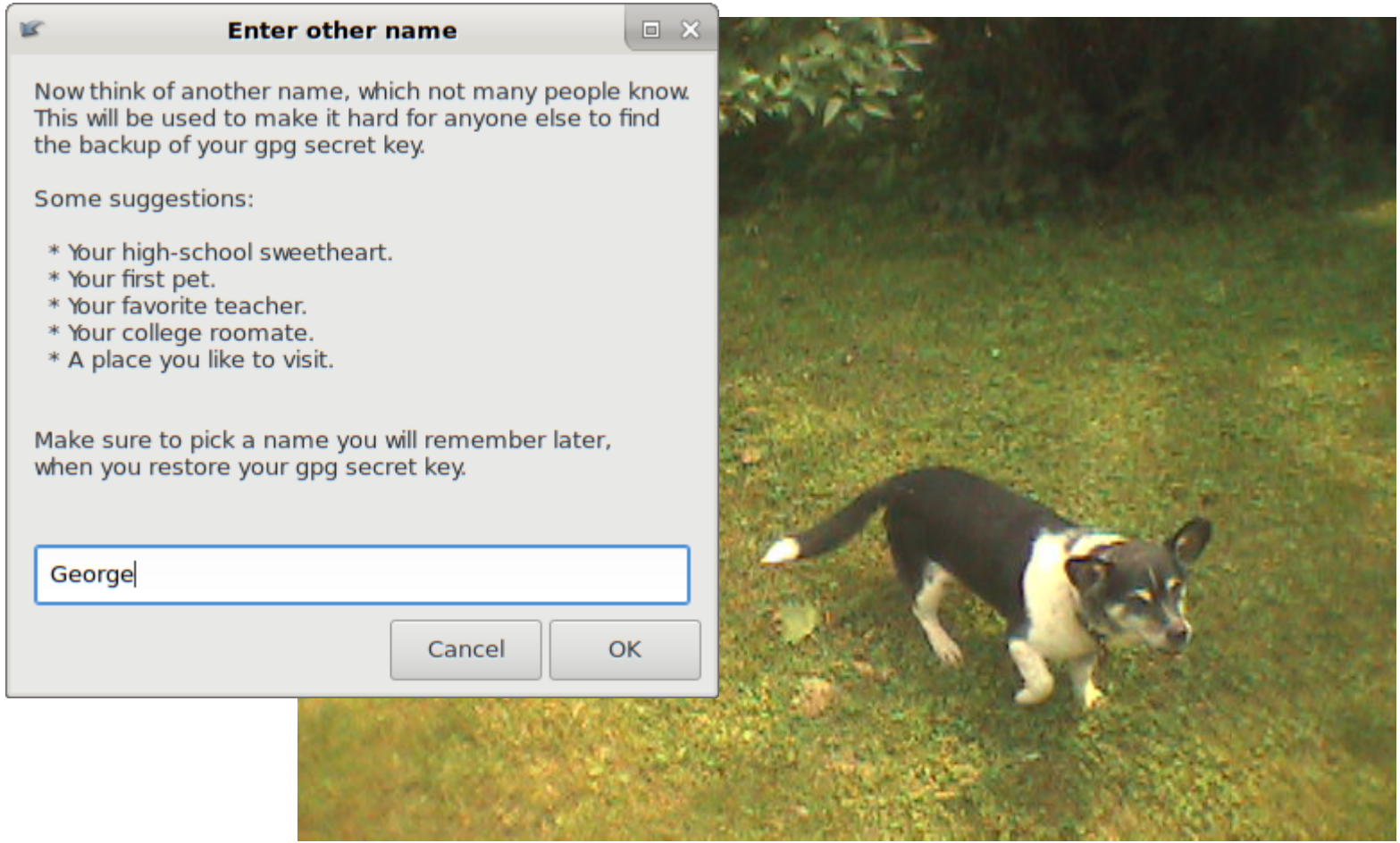
- GPG key backup to cloud servers
- Securely
- Easily



keysafe backup (1/4)



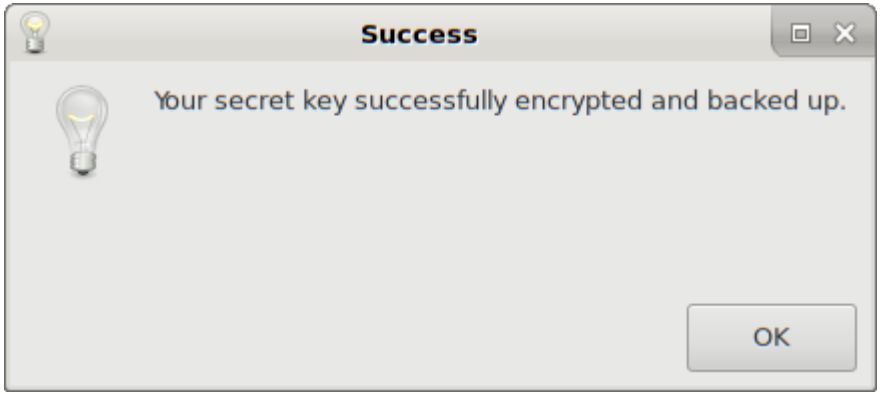
keysafe backup (2/4)



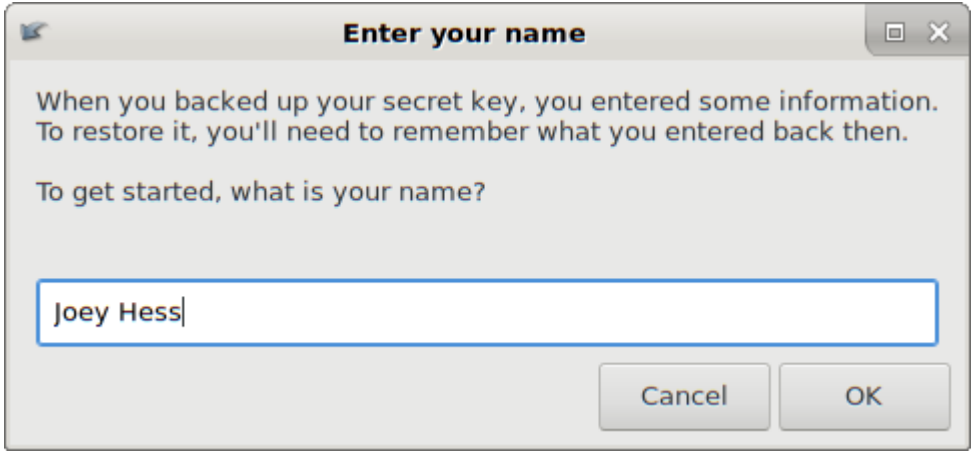
keysafe backup (3/4)

keysafe backup (3/4)

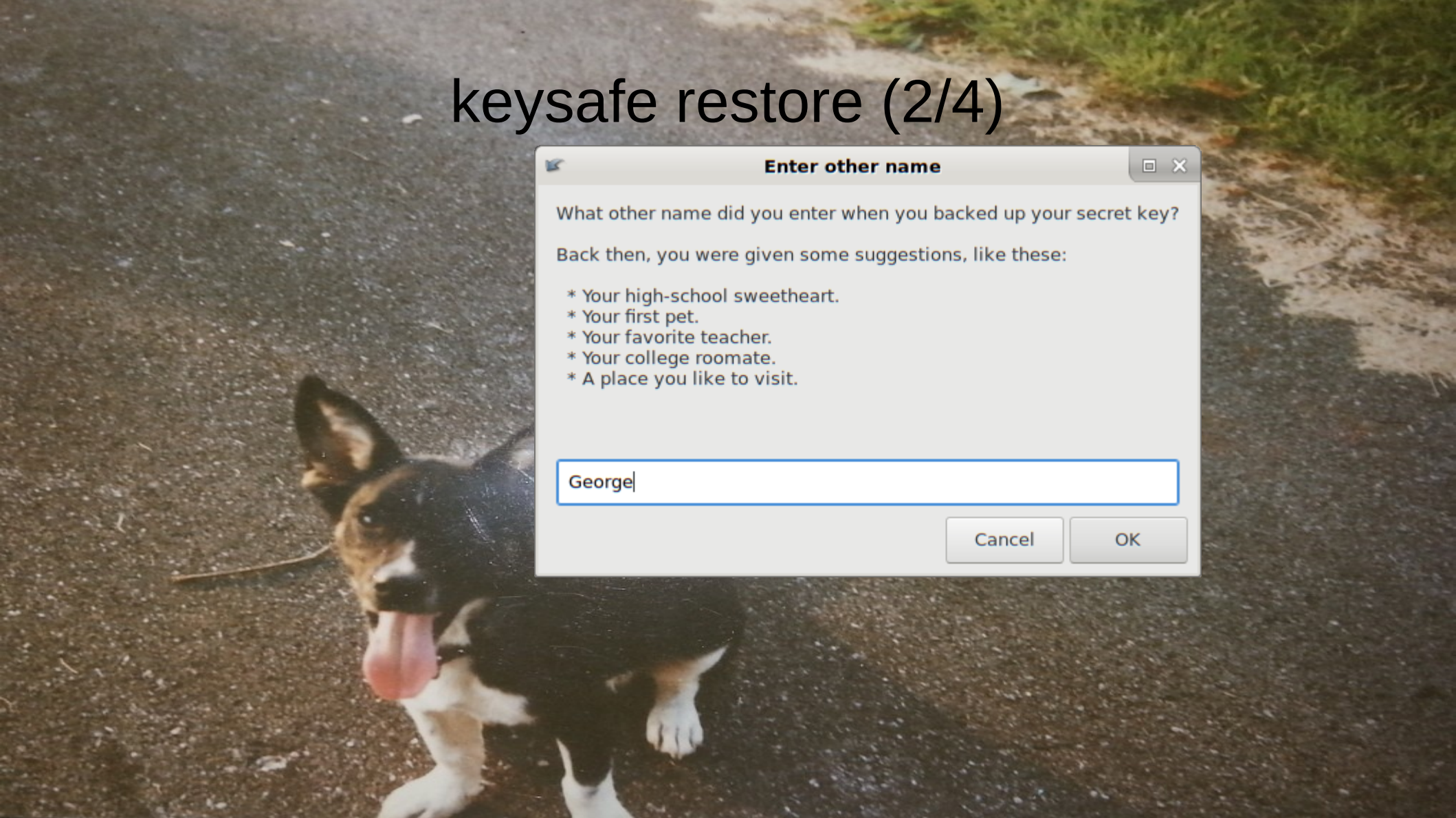
keysafe backup (4/4)



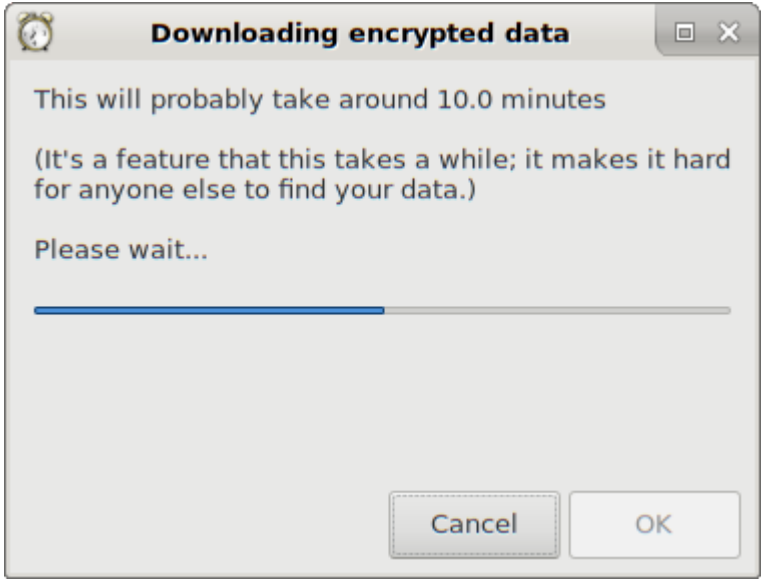
keysafe restore (1/4)



keysafe restore (2/4)



keysafe restore (3/4)



keysafe restore (4/4)

- Wait 25 minutes to 1 hour for decryption...

keysafe’s building blocks

- argon2
- Shamir Secret Sharing
- AES
- The Cloud
- Tor
- zxcvbn

argon2

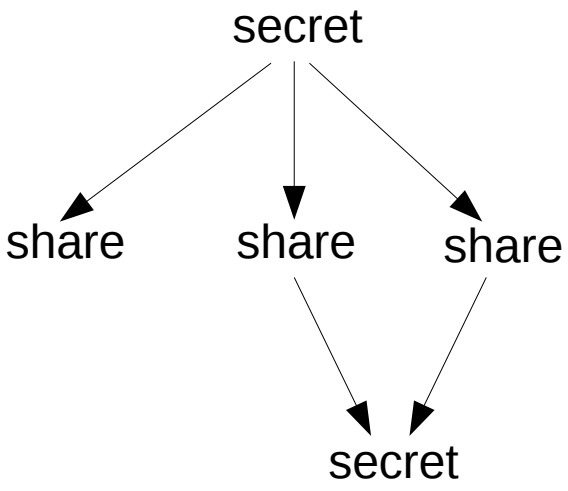
- Password hash
- Password Hashing Competition winner (2015) https://password-hashing.net/

- Memory-Hard
- GPU and ASIC cracking resistance

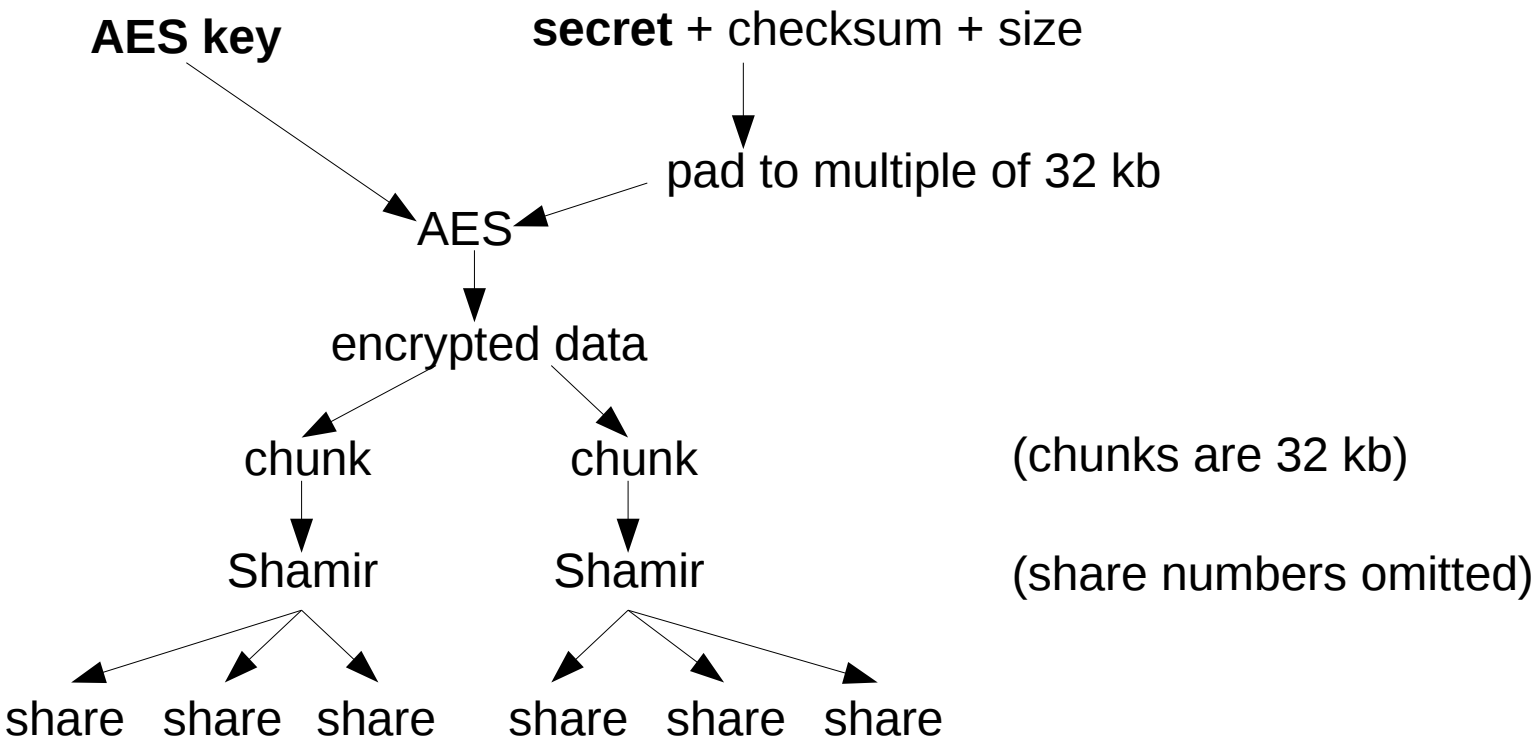
- Tunable difficulty
  - Iterations
  - Memory use
  - Threads

Shamir Secret Sharing

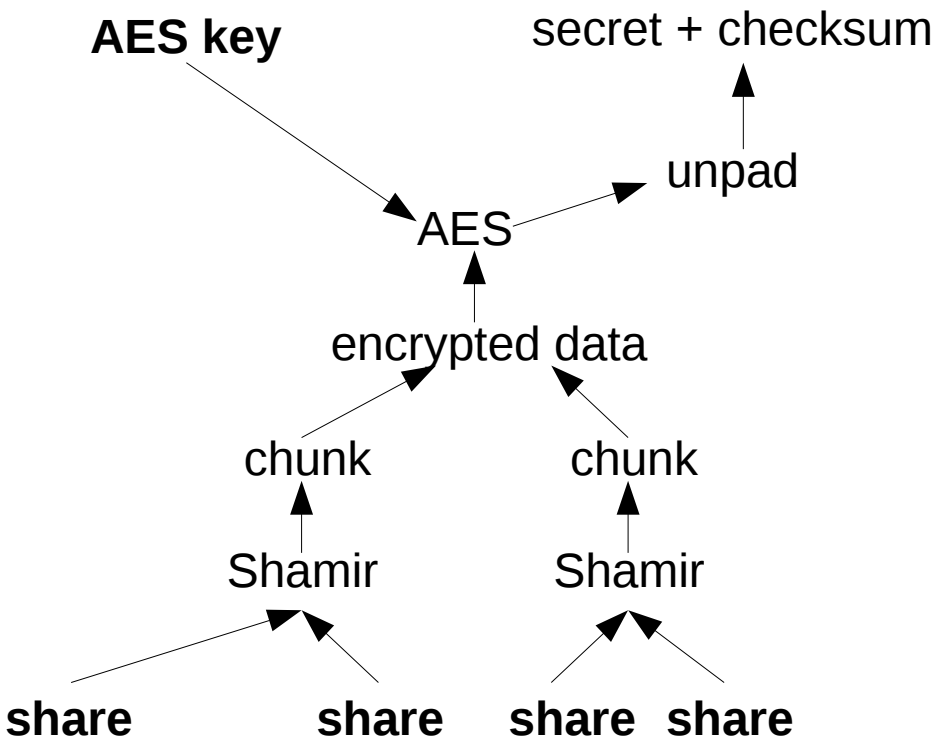
- Boring 70’s technology
- Also completely awesome



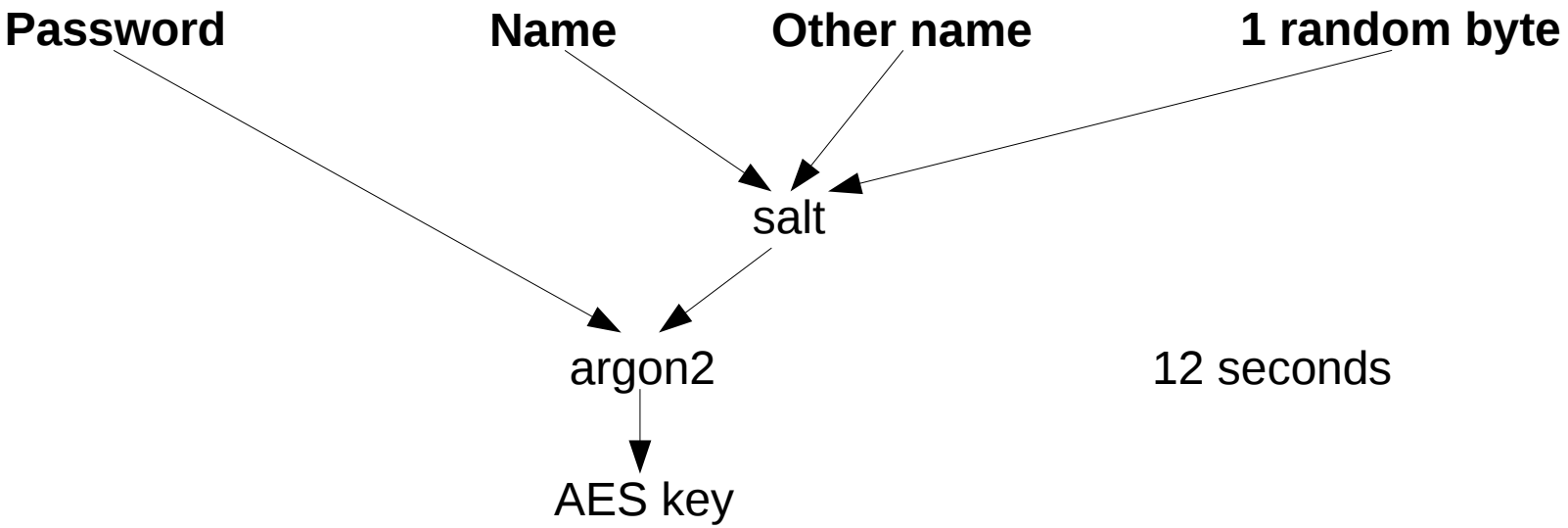
From secret to storable objects



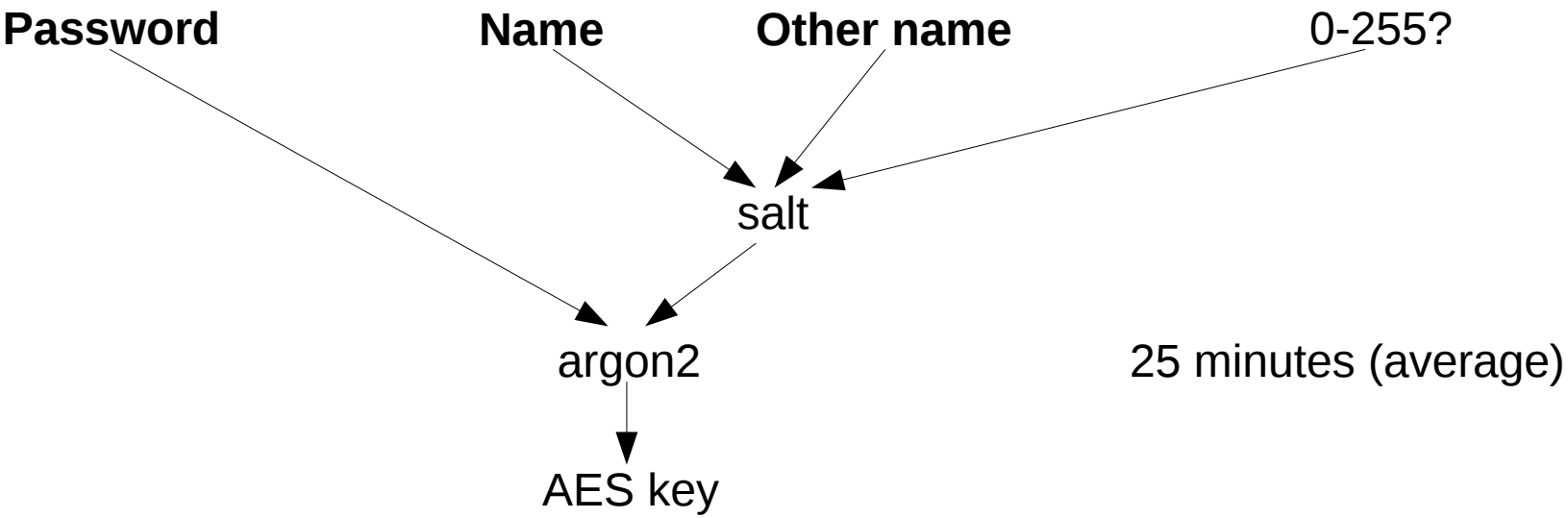
From objects to secret



AES key generation



AES key re-generation

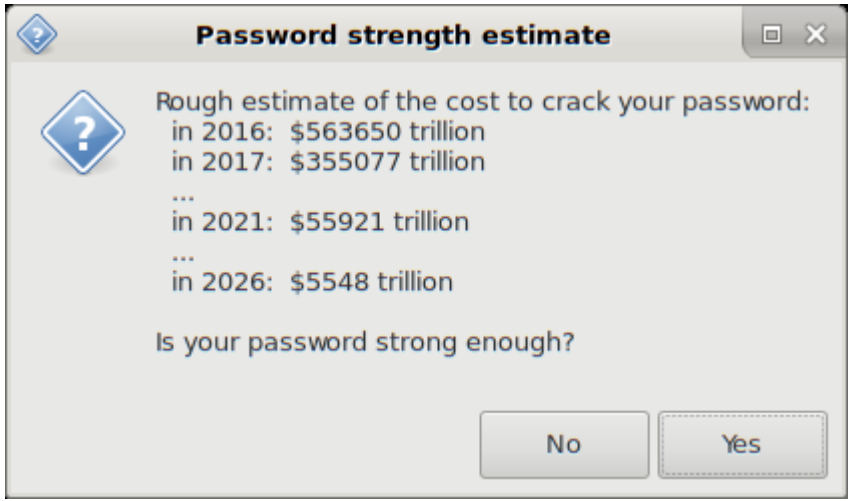


25 minutes (average)

Password cracking cost

- 50 minutes work per guess to generate all 256 possible AES keys

- Weak password (30 entropy) 51072 CPU-years
- Bad password (19 entropy) 25 CPU-years

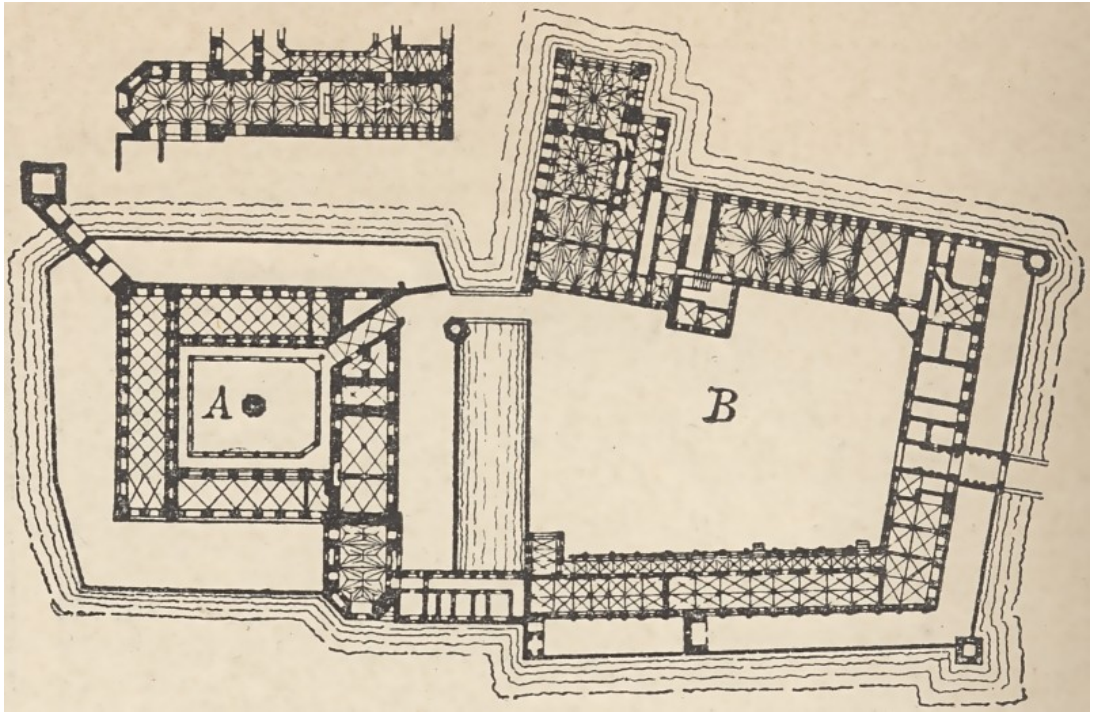


Defenses

A. Password

B. Object IDs

C. Keysafe servers



keysafe servers

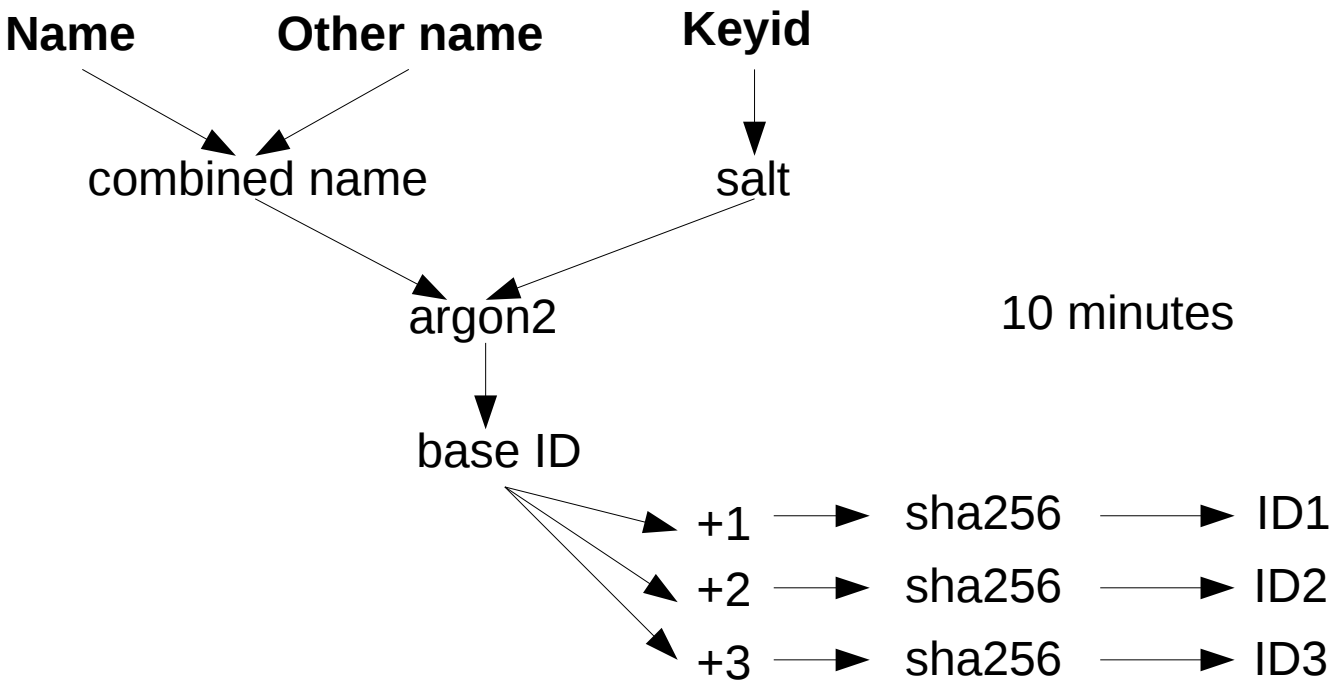
- Store only fixed size objects (no large data)
- Store an object by ID
- Retrieve object by ID
- No object ID enumeration
- Self-tuning proof of work to access
- Accessible only via Tor

keysafe servers

- Other server requirements and best practices (warrant canary)
  https://joeyh.name/code/keysafe/servers/

- As long as 2 of 3 keysafe servers are uncompromised, no mass password cracking.
- Best hosted by well-known, broadly trusted organizations.

Object ID generation



Object IDs

- Attacker needs object IDs to download objects from servers
- Each name guess takes 10 minutes CPU time to calculate object IDs

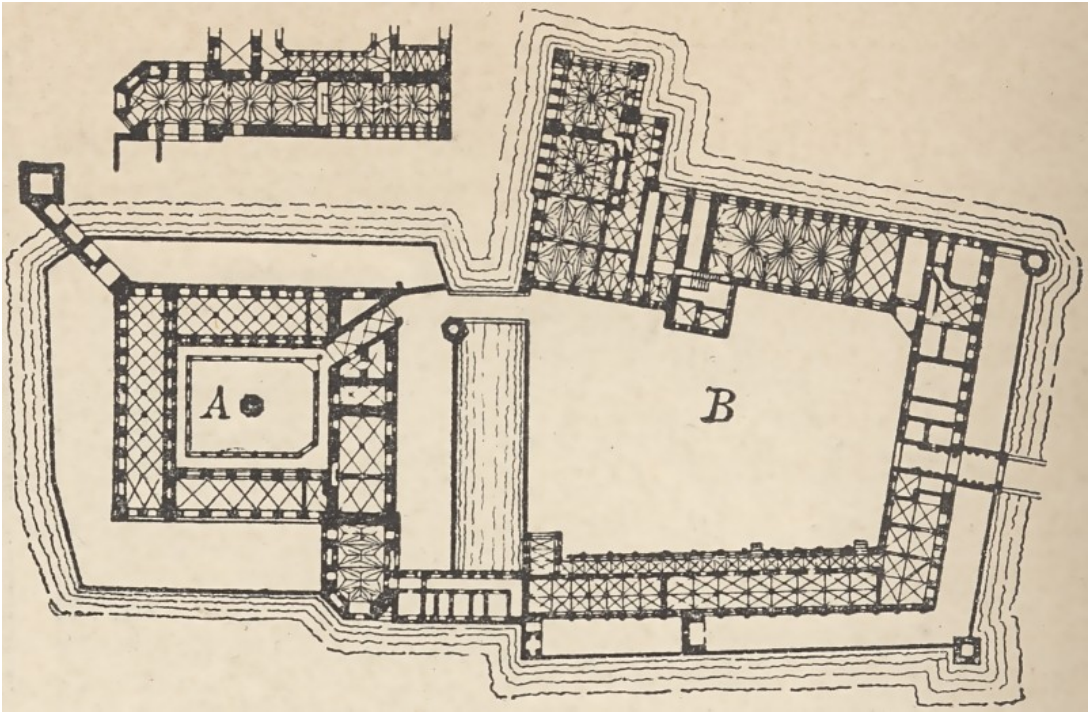
- Two colluding servers can perform a correlation attack to find related object IDs
- Servers don’t record timestamps, or keep logs, to prevent correlation attacks after the fact

Current status

- keysafe client and server implementation in Haskell (3600 LoC)
- In Debian (experimental)
- Needs more design and implementation security review

- Three keysafe servers
  1) Purism
  2) Faelix
  3) Mine at Digital Ocean
- More servers needed

Is keysafe safe enough?



Option for the more paranoid

- Generate 6 shares, with 4 shares needed to recover GPG key
- Store 3 on keysafe servers
- Store 3 locally

- 1 local share + 3 from servers
- 3 local shares + 1 from server

- 64kb share can be stored locally in a variety of hard to detect ways
- End of partition
- Stenanography

Future proofing keysafe

- Decisions, decisions
  - argon2 tuned to take 12 seconds on modern hardware
  - argon2 tuned to take 10 minutes on modern hardware
  - Shamir with 2 of 3 shares
  - 1 byte random salt
  - AES 256 CBC
- May need to change in future in a new version
- Version number metadata would allow partitioning shards
- Solution: Varry object ID generation argon2 memory use parameter depending on version

keysafe

https://joeyh.name/code/keysafe/

Thanks

Purism

https://patreon.com/joeyh