# BULDING A DIY ZERO-TRUST SSH CA

## SECURE AND TRANSPARENT SSH ACCESS MANAGEMENT WITHOUT BLOAT

silent signal

**András Veres-Szentkirályi 2021-08-28**

# $ whoami

**András Veres-Szentkirályi**

- OSCP, GWAPT, SISE
- Silent Signal co-founder
- pentester, toolmaker

# Fahrplan

1. The basics
2. The problem
3. Our solution
4. Final thoughts

# SSH in a nutshell

- complex protocol standardized in RFCs
- PFS encryption, server authentication
- multiple authentication methods

# Hardware tokens

- can be cheap
  - or even “free” like Krypton
- can be “something you have” in 2FA
  - and even enforce “something you know”…
  - …and/or “something you are”
- can be used in various ways
  - resulting in different security levels
- can be lost
  - more on that later
- can be standardized
  - PIV, OpenPGP, FIDO, FIDO2, CTAP, U2F…

# Fahrplan

1. The basics
2. The problem
3. Our solution
4. Final thoughts

# Authentication

- small non-tech organizations and personal servers
  - few servers to log into
  - few users to log in
  - manual tinkering works great
- big organizations
  - SSO
  - dedicated support for this SPoF
- problems for those between the above two
  - technical users
  - revocation
  - tokens

# SSH and hardware tokens

- YubiKey OTP → DEMO$_1$
  - easy to manage, compatible with everything
  - not so secure (think MITM)
- SSH public key authentication → DEMO$_2$
  - more secure (no MITM possible)
  - technical users can be limited (see `AUTHORIZED_KEYS` in sshd(8))
  - who manages the keys? (see `AuthorizedKeysCommand`)
  - public key can come from anywhere (file or device)
  - can use PKCS#11
  - GnuPG offers SSH agent emulation
  - no expiration
- SSH certificates

# SSH certificates

- certificate: issuer signs a statement about a subject’s public key
- SSH certificate: much simpler than X.509
  - simple serialization format
  - no multi-layer PKI implemented
- has expiration, can be revoked
- can have limitations (e.g. which commands can be executed)
- lots of trust placed in CA(s)
- much less supported than “plain” public key authentication
  - OpenSSH supports a lot, yet not everything
  - most other clients – not so much
  - OpenSSH example: port forward granularity

# SSH certificate authentication

- TrustedUserCAKeys: like authorized_keys, just for CAs
- Principal: list of strings
  - can be a literal username → $DEMO_3$
  - can match an entry in AuthorizedPrincipalsFile
- AuthorizedPrincipalsCommand: taking it to the next level, like with keys
- RevokedKeys: refuses otherwise valid certificates

# CA trust and transparency

- has the CA signed a certificate it shouldn’t have?
- can the CA demonstrate that its key is secure?
- do leaf certificates match the policy?
  - expiration date
  - key security
  - limitations
- what to do if something has gone wrong?
  - compromised CA
  - compromised user key
  - improperly issued certificates
  - destroyed/lost tokens

# Fahrplan

1. The basics
2. The problem
3. Our solution
4. Final thoughts

# Attestation

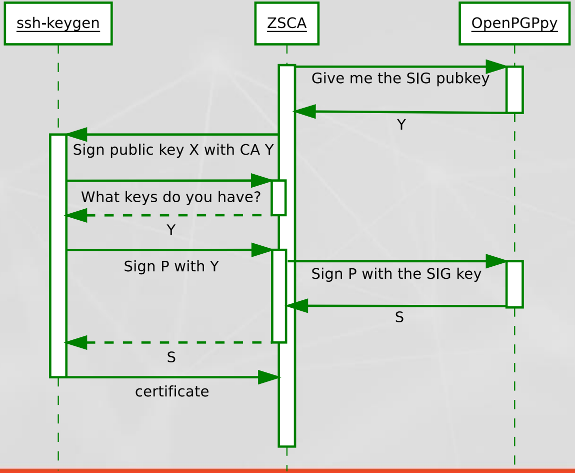
“The concept of attestation is to cryptographically certify that a certain asymmetric key has been generated on device, and not imported. This can be used to prove that no other copies of the asymmetric key exist.” – `https://developers.yubico.com/PGP/Attestation.html`

- the implementation is YubiKey-specific, but the idea is not
- X.509 both for PIV and OpenPGP
- can be parsed with OpenSSL (→ DEMO$_4$) and `https://cryptography.io/`
- **our take: necessary for regular users and CAs**

# OpenPGP ($\neq$ GnuPG)

- supports EdDSA (Ed25519) on newer YubiKeys
  - unlike PIV, which supports RSA and ECDSA only
- subpar everyday UX
  - unlike PIV, which has `https://github.com/FiloSottile/yubikey-agent`
- has a signature counter → $\text{DEMO}_5$
  - but only for the signing key, not the (technically identical) authentication key
  - GnuPG SSH agent emulation can only use latter
- besides GnuPG, there's a low-level Python implementation
  - `https://github.com/bitlogik/OpenPGPpy` → $\text{DEMO}_6$
  - Ed25519 had problems, see issue #1
- **our take: signature counter is a must-have for CAs**

# How it all works together



# Attacker model

- attacker can make the CA sign something it shouldn't have
- if it gets saved into the database, it can be seen during an audit
- if it's not in the database, counter doesn't match the number of certs
- centralized logging and SIEM could improve this even further

"Testing shows the presence, not the absence of bugs" – Dijkstra (1969) J.N. Buxton and B. Randell, eds, Software Engineering Techniques, April 1970, p. 16. Report on a conference sponsored by the NATO Science Committee, Rome, Italy, 27–31 October 1969.
`http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1969.PDF`

# Further tests

- every attestation chain is valid
- every attestation leaf certificate indicate hw-generated keys
- every attestation leaf certificate matches the unique Yubikey ID
- every SSH certificate is valid and unique
  - the public key within the certificate matches that of the *Pubkey*
  - the signature is can be verified using the *Pubkey* of the *CA*
  - the certificates differ in at least 1 bit, thus their signature differs as well, proving that the signature counter was incremented
- every SSH certificate has an expiration date within a preconfigured limit

# Fahrplan

1. The basics
2. The problem
3. Our solution
4. Final thoughts

# The result

- “Look ma, no secrets!”
- anyone can inspect the database and verify its integrity
- currently Python/Django
  - nothing specific to these stacks
  - could be implemented in anything else
  - we already have it in the stack and the libraries were nice
- many hate PGP…but we use nothing (OpenPGP serialization, GnuPG tools, keyservers, web-of-trust) that this hatred is focused on
- many hate certificates…but we use nothing (X.509 and thus ASN.1, sub-CAs) that this hatred is focused on

# Future plans

- web interface (Django makes this easy)
- self-service renewal
- handle first three PGP (self-)signatures

# Outro

- source code and binaries under MIT: `https://github.com/silentsignal/zsca`
- core functionality WORKSFORME
- pull requests welcome
- we're hiring!

# THANKS!

**ANDRÁS VERES-SZENTKIRÁLYI**

vsza@silentsignal.hu

facebook.com/silentsignal.hu

@SilentSignalHU

@dn3t

silent signal