OpenSSH public/private key login

risico's en mitigatie

Wie ben ik…?

12-09-1969

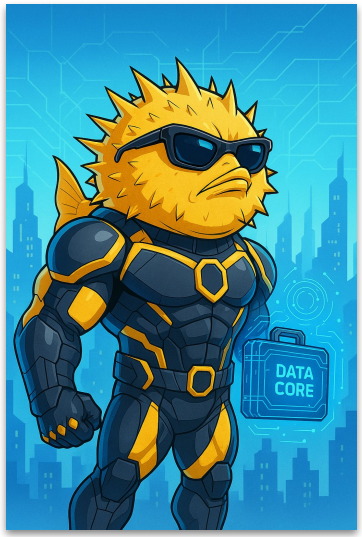
Winfried de Heiden
w.d.heiden@atcomputing.nl
Linux/Open Source Consultant & Trainer bij AT Computing
Security liefhebber:

- smart cards
- passkeys
- tpm2
- hekel aan wachtwoorden…

eS$<IXg0MZ^qZU)Ft}Ef

OpenSSH Public Key Login

- OpenSSH public key login:
  - Sinds 1999
  - Cryptografisch sleutelpaar: 1x privé sleutel + 1x publieke sleutel
  - Plaats publieke sleutel op server
  - Login is alleen mogelijk bij bezit privé sleutel
  - Lastiger te kraken dan wachtwoorden
  - Inloggen-zonder-wachtwoord
  - OpenSSH public key login super hero…?



Doel / agenda

- Zwakheden public/private key login & mitigatie
  - Verkeerd en/of malafide gebruik publieke sleutels
  - Verkeerd en/of malafide gebruik privé sleutels

- Waarom dan?
  - De toekomst is zonder wachtwoorden
  - Veiliger gebruik public/private key login

- (mijn) Uitgangspunten:
    - Grote (enterprise) omgeving
    - Controleerbaar (audits!)
    - Zo veel mogelijk standaard & open source software

Verkeerd/malafide gebruik publieke sleutels

- Standaard locatie authorized_keys:

```
~/.ssh/authorized_keys
```

- "gewoon" bestand, beheerd door eindgebruikers:
  - Account delen met collega ("kun je tijdens mijn vakantie…")
  - Malafide toevoeging ander account. (malware, insider threat, browser plugin, …)
  - Geen controle ~/.ssh/authorized_keys

Wanneer heb jij voor het laatst ~/.ssh/authorized_keys gecontroleerd?

Verkeerd/malafide gebruik publieke sleutels

- Mitigatie 1 - **immutable** ~/.ssh/authorized_keys:

```
$ sudo chattr +i ~userx/.ssh/authorized_keys
$ lsattr ~userx/.ssh/authorized_keys/authorized_keys
----i----------------- authorized_keys
```

quick-n-dirty maar effectief!

Verkeerd/malafide gebruik publieke sleutels

- Mitigatie 2: authorized_keys op **andere locatie**:

```
## /etc/ssh/sshd_config

AuthorizedKeysFile /etc/security/sshkeys/%u/authorized_keys
```

  - Leesbaar voor (login) user
  - Schrijfbaar voor (typisch) root
  - Controle authorized_keys niet langer bij eindgebruiker.

Verkeerd/malafide gebruik publieke sleutels

- Mitigatie 3: **centrale locatie** authorized_keys:
  - webserver
  - (read only) NFS
  - LDAP
  - …
  - Download de public key met script, zelf-bouw software
  - Standaard bij FreeIPA en FreeIPA clients

authorized_keys - Git Managed

- **Git** als centrale *backend store* SSH user public keys.
  - Git vaak al aanwezig
  - Git biedt toegangscontrole en autorisaties
  - Protected branches geven controle
  - Creëer public-key-download-script in je eigen favoriete programmeertaal:


```
## /etc/ssh/sshd_config

Match user root,demouser1,demouser2
     AuthorizedKeysCommand /usr/local/bin/get_ssh_pubkey.sh %u
     AuthorizedKeysCommandUser nobody
     AuthorizedKeysFile none
```

lokale authorized_keys niet gebruikt.

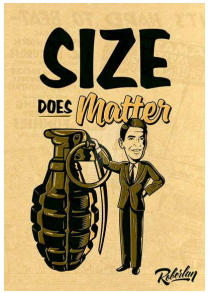
Verkeerd/malafide gebruik privé sleutels

- Gebruik zwakke sleutels (DSA, te kort…)
- Mitigatie zwakke sleutels (beperkt) mogelijk:

```
## /etc/ssh/sshd_config

RequiredRSASize 3072
```

Alleen voor RSA keys



- Elliptic Curve (**ecdsa**) en Edwards-curve Digital Signature Algorithm (**ed25519**)
  - Korter met behoud van veiligheid

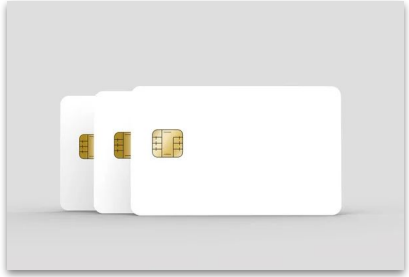
Verkeerd/malafide gebruik privé sleutels

- Privé sleutel, "gewone bestanden", lekken weg:
  - Onveilige opslag (shares, Google Drive, Git, email, usb drives, …)
  - Aantal kopieën **niet traceerbaar**
  - Privé sleutel *met of zonder* wachtwoord: **niet controleerbaar**
  - Onderling delen van privé sleutels: **niet controleerbaar**
  - Diefstal d.m.v. kopie: (vrijwel) **niet controleerbaar**

certificaatsleutels: moeilijk kopieerbaar

Verkeerd/malafide gebruik privé sleutels

- Mitigatie "rondslingerende" privé sleutels:
  - FIDO2 (Yubikey, Nitrokey, Google Titan, …)
    - OpenSSH >= 8.2
    - <= EL8: niet ondersteund
  - Chip cards - smart cards - PKCS #11 cards
    - (min of meer) standaard en aanwezig in EL
    - Relatief goedkoop



PKCS #11 cards

- PKCS #11:
  - Standaard API naar *devices* met cryptografische informatie
  - Bijvoorbeeld smart cards
  - Opslag van bijvoorbeeld privé sleutels

- Opensource:
  - OpenSC: https://github.com/OpenSC/OpenSC/wiki
  - Standaard beschikbaar in veel Linux distributies

- Bekend terrein (soms):
  - Veel organisatie gebruiken al smart cards
  - Aanhaken op bestaande infrastructuur
  - Fysieke toegang, Windows toegang, Linux toegang etc.

Kaartlezers

- Kaartlezer
  - Ingebouwd in laptop
  - Toetsenbord met kaartlezer
  - USB kaartlezer

- Ondersteund op veelal pcsc-lite - https://pcsclite.apdu.fr
  - Veel is ondersteund - https://ccid.apdu.fr/ccid/supported.html
  - Maar veel ook niet :(
  - Leveranciers, fabrikanten: onduidelijke specificaties

PKCS #11 smart cards

- Kaartlezers was het gemakkelijke deel
- Smart Cards lastiger…

- Uitzoekwerk:
  - https://github.com/OpenSC/OpenSC/wiki/Supported-hardware-(smart-cards-and-USB-tokens)
  - Onduidelijk welke smartcard waar verkrijgbaar is
  - Beschrijving niet altijd actueel
  - Onduidelijke specificaties
  - Geen of gebrekkige ondersteuning Linux

OpenSC: ook voor Windows!

PKCS #11 smart cards

- Software:

```
# dnf install   pcsc-lite opensc gnutls-utils pcsc-tools
```

- Smart Card reader:
  - Kaartlezer (bijna altijd usb) herkend?

```
$ lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 08e6:4433 Gemalto (was Gemplus) GemPC433-Swap
Bus 001 Device 003: ID 0627:0001 Adomax Technology Co., Ltd QEMU Tablet
Bus 001 Device 004: ID 072f:b100 Advanced Card Systems, Ltd ACR39U
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
```

PKCS #11 smart cards

- Kaartlezer herkend?

```
$ pcsc_scan

PC/SC device scanner
V 1.6.2 (c) 2001-2022, Ludovic Rousseau <ludovic.rousseau@free.fr>
Using reader plug'n play mechanism
Scanning present readers...
0: ACS ACR39U ICC Reader 01 00

Fri Apr 11 17:55:35 2025
 Reader 1: ACS ACR39U ICC Reader 01 00
  Card state: Card inserted,
  ATR: 3B F5 96 00 00 81 31 FE 45 4D 79 45 49 44 14
~
```

PKCS #11 smart cards

- Smart Card herkend?

```
$ pcsc_scan

~
Possibly identified card (using /usr/share/pcsc/smartcard_list.txt):
3B F5 96 00 00 81 31 FE 45 4D 79 45 49 44 14
3B F5 96 00 00 8. 31 FE 45 4D 79 45 49 44 1.
     MyEID card (Infineon chip) (PKI)
     https://services.aventra.fi/English/products_MyEID_E.php
```

Kaart en kaartlezer herkend!

PKCS #11 smart cards

- Privé sleutel op kaart: twee opties:

- (1) Bestaande privé sleutel op kaart plaatsen:
  - Nieuwe kaart, bestaand certificaat is mogelijk, maar…
  - …veilige opslag privé sleutels noodzakelijk

- (2) Nieuwe privé sleutel op kaart genereren:
  - Eén sleutel = één kaart
  - Geen opslag privé sleutels = extra veilig
  - Kaart weg = pech
  - **Tip:** minimaal 2 kaarten (= 2 privé sleutels = 2 publieke sleutels)

PKCS #11 smart cards

- Privé sleutel op kaart genereren:

```
$ pkcs15-init --erase-card --use-default-transport-keys

$ pkcs15-init --create-pkcs15 --use-default-transport-keys --pin 123456 \
  --puk 123456 --so-pin 12345678 --so-puk 123456

$ pkcs15-init --store-pin --label userx --auth-id 01 --so-pin 12345678 \
  --pin 123456 --puk 123456

$ pkcs15-init --generate-key ec:secp521r1 --auth-id 01 --so-pin 12345678 \
  --pin 123456 --label userx_key

$ pkcs15-init --finalize
```

Elliptic Curve sleutel

PKCS #11 smart cards

- Publieke sleutel van kaart lezen:

het werkt!

```
$ sudo ssh-keygen -D pkcs11:
ecdsa-sha2-nistp521
AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAEgLrXSicR0Kd8rQEudeUjM/Q
V8vV6VzabYA3h9NFGQD4epYNQQULqb2t44jWSCRyxCjUufINBK9LVe0q0wrroEPAFuut/oL5FnQ5
~
```

- Alternatief:

```
$ sudo ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so
```

nu nog als user root

Polkit en kaartlezers

- Kaartlezers toegang:
  - PolKit blokkeert standaard behalve root
  - Verbeterde veiligheid: niet alle processen krijgen toegang tot kaart = privé sleutel
  - Creëer groep voor smart card toegang…

```
$ id userx
uid=1000(userx) gid=1000(userx) groups=1000(userx),10(wheel),986(smartcard)
```

  - …en PolKit rule

Polkit en kaartlezers

```
## /etc/polkit-1/rules.d/01-smart-card.rules

[root@kaartlezer-server rules.d]# cat
/etc/polkit-1/rules.d/01-smart-card.rules
polkit.addRule(function(action, subject) {
    if (action.id == "org.debian.pcsc-lite.access_card" &&
        subject.isInGroup("smartcard")) {
            return polkit.Result.YES;
    }
});
polkit.addRule(function(action, subject) {
    if (action.id == "org.debian.pcsc-lite.access_pcsc" &&
        subject.isInGroup("smartcard")) {
            return polkit.Result.YES;
    }
});
```

leden van de groep smartcard krijgen toegang tot lezer en kaart

PKCS #11 - kaart login

- Gebruik pkcs #11 voor login:

```
$ ssh -i pkcs11: demouser1@kaartlezer-client
Enter PIN for 'MyEID (userx)':
```

- Soms noodzakelijk: verwijs naar specifieke pkcs #11 library:

```
$ ssh -I /usr/lib64/pkcs11/opensc-pkcs11.so demouser1@kaartlezer-client
Enter PIN for 'MyEID (userx)':
```

- PIN = wachtwoord op privé sleutel
- Wachtwoordloos…?

Windows 11 & Windows Terminal:
werking vrijwel identiek!

PKCS #11 - ssh-agent

- PKCS #11 wachtwoord? Gebruik de ssh-agent!
  - Systemd user service ssh-agent.service
  - Standaard aanwezig in EL9

start ssh-agent indien niet actief

```
## ~/.bash_profile

systemctl --user start ssh-agent.service
export SSH_AUTH_SOCK="/run/user/$(id -u)/ssh-agent.socket"
```

zelfde socket = zelfde agent

PKCS #11 - ssh-agent

- Systemd user ssh-agent:

```
$ ssh-add pkcs11:
Enter passphrase for PKCS#11:
Card added: pkcs11:
```

```
$ ssh-add -L
ecdsa-sha2-nistp521
AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAAIbmlzdHA1MjEAAACFBAEgLrXSicR0Kd8rQEudeUjM/
~
```

```
$ ssh demouser1@kaartlezer-client 'hostname'
kaartlezer-client.demo.lan
```

geen wachtwoord!

PKCS #11 - ssh-agent

- Login remote server:

```
$ ssh demouser1@kaartlezer-client
$ ls ~/.ssh/authorized_keys
ls: cannot access '/home/demouser1/.ssh/authorized_keys': No such file …
```

  - ~/.ssh/authorized_keys niet aanwezig
  - publieke sleutel uit git!

```
## /etc/ssh/sshd_config

Match user root,demouser1,demouser2
      AuthorizedKeysCommand /usr/local/bin/get_ssh_pubkey.sh %u
      AuthorizedKeysCommandUser nobody
      AuthorizedKeysFile none
```

Samenvatting

- OpenSSH public/private key login zwakheden mitigeren:
  - Publieke sleutels zijn publiek…
  - …maar **~/.ssh/authorized_keys** vraagt aandacht

- Houd privé sleutels privé/geheim:
  - Voorkom "rondslingerende" privé sleutels
  - Gebruik veilige opslag, bijv. smart cards

public/private key login

Einde

Vragen?