Yubikey

Discovery and first use of Yubico's Yubikey

Presented by :

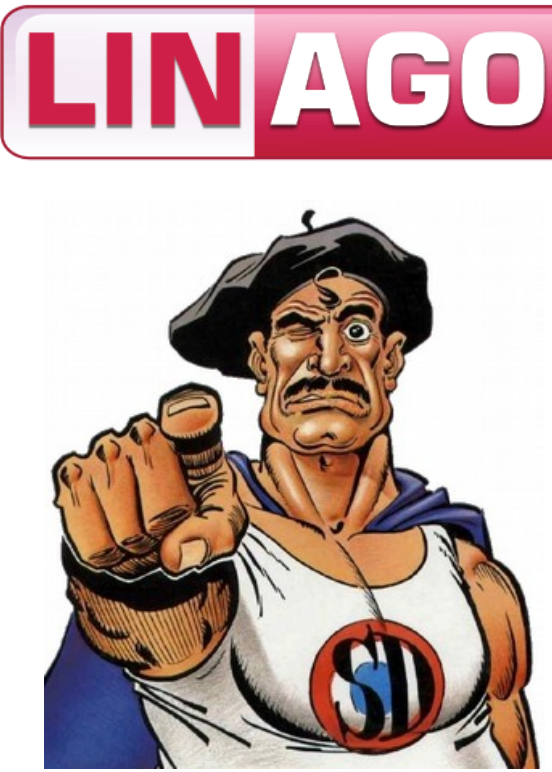
Maxime de Roucy

mderoucy@linagora.com – http://dokuwiki.craoc.fr/

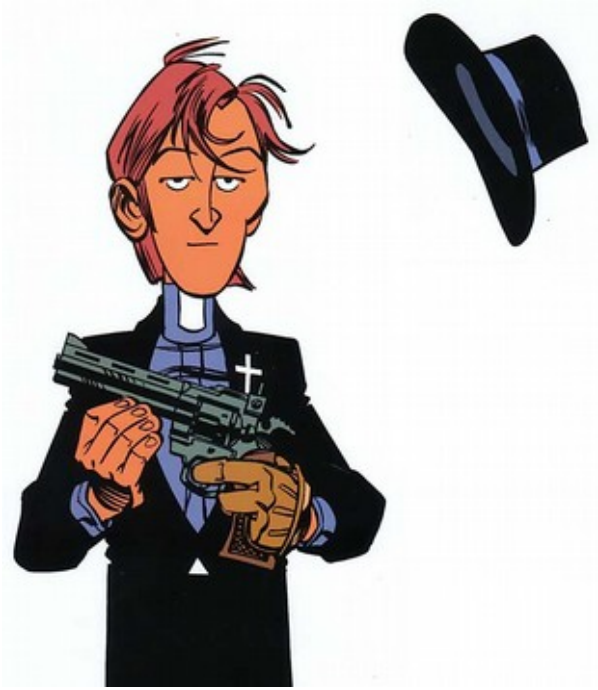
RMLL MONTPELLIER 2014

Le libre et vous !
15èmes Rencontres Mondiales du Logiciel Libre
Du 5 au 11 juillet 2014

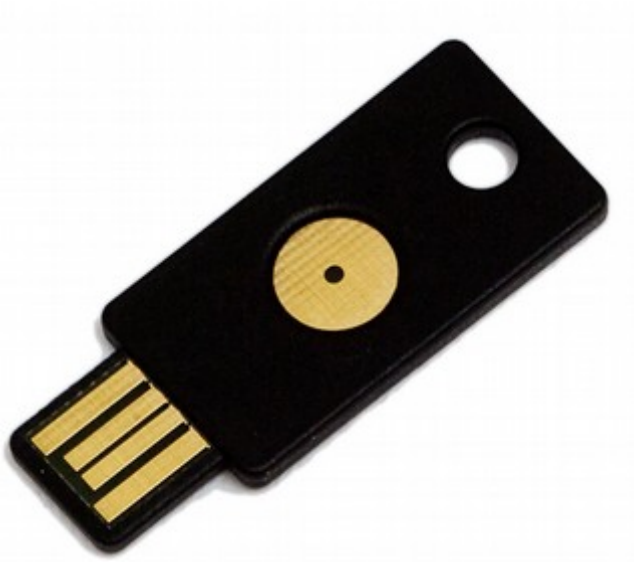
About myself (really quick I promise)



- Job
  - Technical Account Manager
  - OSSA
    - Open Source Software Assurance
  - Linagora
- Sports
  - swimming
  - inline skating
- Geek
  - Linux (Archlinux, Gentoo)
  - comic books (Bourgeon, Tome)



What's a Yubikey



- A yubikey is an authentication USB device
- sold by the Yubico company
- detected as standard keyboard
- open source softwares (servers, modules…)
- generate One Time Password (OTP)
- several security algorithm can be chosen
- two configuration slots
- can be software triggered (Challenge – Response mode)
- no moving parts, mono-block
- protection class : IP 67 (dust tight, waterproof : 1m - 30min)

What will we (try) to talk about ?

- First configuration
  - algorithms
  - modes
- My / Sample configurations
  - PAM modules overview
  - Desktop / Gnome
    - PAM Challenge-Response mode
    - auto-lock session when yubikey is removed
  - Server / SSH
    - Yubiserve OTP authentication server
    - PAM Yubico OTP mode

Algorithms / Modes (1/2)

- Yubico OTP
  - preconfigured OTP (used against Yubico authentication server)
  - public ID (6 B) , Private ID (6 B), secret AES key (16 B = 128 bits)
  - 15bits non-volatile & non-circular counter ; 1 B volatile & circular counter
  - 3B timestamp, 8Hz, random seed
  - 2B Random number (generator input : USB traffic, output touch sensor)
- OATH-HOTP
  - Initiative for open authentication (RFC 4226)
  - Hashed One Time Password
  - KeePass, ~ Google ~
  - same non-volatile & non-circular counter

Algorithms / Modes (2/2)

- Static Password (if we have the time ☺)
- Challenge-Response
  - can be configured to require user interaction
  - Yubico OTP
    - use the counter
    - 6 Bytes challenge (XORed with the private ID)
    - different output for the same challenge
  - HMAC-SHA1 (RFC 2104)
    - don't use the counter
    - 0-64 Bytes challenge, 20 Bytes secret
    - same output for the same challenge

During this talk

- Configure first slot Yubico OTP
- Configure second slot with Challenge-Response HMAC-SHA1

```
netbook % cat yubi.log
LOGGING START,08/07/2014 21:34
Yubico OTP,08/07/2014
   21:34,1,vvirbtrlvrgn,0912031df04f,bf4f68c1bc1a7ffb16bdf045472b88d9,,
   ,0,0,0,0,0,0,0,0,0,0,0
Challenge-Response: HMAC-SHA1,08/07/2014
   21:47,2,,,ec5f5fd02d9627cbde9d3d7e3ce5fa50ff4eb8b8,,,0,0,0,0,0,0,0,0
   ,0,0
```


PAM modules

- pam_yubico
  - official
  - Archlinux, AUR : yubico-pam-git, pam_yubico
  - Gentoo : sys-auth/pam_yubico
  - online validation : Yubico OTP
    - possibility to use your own validation server
  - offline validation : Challenge-response MAC-SHA1
- yubipam
  - offline validation : Yubico OTP

Desktop

- /etc/pam.d/system-auth

```
desktop % diff -u /etc/pam.d/system-auth{.save,}
[…]
+auth        required  pam_env.so
+auth        sufficient pam_yubico.so mode=challenge-response
 auth        required  pam_unix.so     try_first_pass nullok
 auth        optional  pam_permit.so
-auth        required  pam_env.so
[…]
```

  - use `required` instead of `sufficient` for two-factor authentication

- record the plugged yubikey (C/R HMAC-SHA1 configured on slot 2)

```
desktop % ykpamcfg -v -2
desktop % stat ~/.yubico/challenge-1620890
Accès : (0600/-rw-------)  UID : ( 1000/max)   GID : (  100/users)
```

- tty, gdm, sudo… plug your yubikey → no need to enter password anymore

Gnome (1/2)

- Unlock gnome-keyring-daemon if we use password
- don't start gnome-keyring-daemon if we use the yubikey
  - gnome-keyring-daemon will start and ask for password at first need

```
desktop % diff gdm-password{.save,}
2c2
< auth     optional  pam_gnome_keyring.so
---
> auth     optional  pam_gnome_keyring.so auto_start
11c11
< session  optional  pam_gnome_keyring.so auto_start
---
> session  optional  pam_gnome_keyring.so
```


Gnome (2/2)

- auto lock session when a yubikey is unplugged

```
max@max-desktop % cat /etc/udev/rules.d/70-yubikey.rules
ACTION=="remove", SUBSYSTEM=="usb", ENV{ID_VENDOR_ID}=="1050",
   ENV{ID_MODEL_ID}=="0010", RUN+="/usr/local/bin/yubikey-gnome-lock"
desktop % sudo cat /usr/local/bin/yubikey-gnome-lock
#! /bin/bash
export DISPLAY=':0'
#su max -c "/usr/bin/gnome-screensaver-command -l"
su max -c "dbus-send --type=method_call --dest=org.gnome.ScreenSaver
   /org/gnome/ScreenSaver org.gnome.ScreenSaver.Lock"
desktop % sudo stat /usr/local/bin/yubikey-gnome-lock
Accès : (0700/-rwx------)  UID : (0/root)   GID : (0/root)
```

Note : launch a script at plug event

```
ACTION=="add", SUBSYSTEM=="usb", ATTRS{idVendor}=="1050",
   ATTRS{idProduct}=="0010", RUN+="/path/mon_script"
```


Server / SSH

- Gentoo server

Yubiserve

- https://code.google.com/p/yubico-yubiserve/
- python2, sqlite3
- simple & standalone (don't need other service to run)
- support Yubico OTP & OATH-HOTP algorithms
- Gentoo ebuild downloadable from my server
  - ftp://craoc.fr/yubiserve-ebuild.tar.xz

- /etc/yubiserve.cfg

```
yubiservePORT = 8000;
yubiserveSSLPORT = 8001;
yubiserveHOST = '0.0.0.0';
yubiDB = 'sqlite3';
yubiserveDebugLevel = 0;
yubiserveCERT = '/etc/ssl/yubiserve/yubiserve.pem';
```

- generate an API key and its ID (used in /etc/pam.d/sshd)

```
server % sudo yubiserve-dbconf -aa testapikey
New API Key for 'testapikey': 'TVJqNnJHaXNXemUyaW1Jam9mczc='
Your API Key ID is: 2
```

- store your yubikey OTP informations in the database
  - <nickname> <publicid> <secretid> <aeskey>

```
server % sudo yubiserve-dbconf -ya testkey vvirbtrlvrgn 0912031df04f
   bf4f68c1bc1a7ffb16bdf045472b88d9
Key 'testkey' added to database.
```


PAM

- single factor authentication
  - yubikey OTP
  - ssh key
- ~/.yubico/authorized_yubikeys
  - <user name>:<yubikey public ID>:<yubikey public ID>:…

```
max:vvirbtrlvrgn
```

- /etc/pam.d/sshd (pam_unix is disabled)

```
server % diff -u /etc/pam.d/sshd{.save,}
-auth    include   system-remote-login
+#auth   include   system-remote-login
…
+auth    required  pam_env.so
+auth    required  pam_yubico.so id=2 key=TVJqNnJHaXNXemUyaW1Jam9mczc=
  url=http://127.0.0.1:8000/wsapi/2.0/verify?id=%d&otp=%s mode=client
+auth    optional  pam_permit.so
```

- /etc/sshd/sshd_config

```
PasswordAuthentication yes
ChallengeResponseAuthentication no
UsePAM yes
```

- test

```
server % sed -i 's/^\(.*max-netbook\)/#\1/' .ssh/authorized_keys
Connection to max-server closed.
netbook % ssh max-server
max@max-server's password:
server % sed -i 's/^#\(.*max-netbook\)/\1/' .ssh/authorized_keys
Connection to max-server closed.
netbook % ssh max-server
server % ☺
```


Static password

- yubikey-personalization-gui
  - OK for most “standard” keyboard layout (qwerty, azerty…)
  - builtin mapping between characters & scancode
- ykpersonalize
  - more complex but powerful

non standard layout

```
netbook % sudo getscancodes /dev/input/by-id/usb-
   TypeMatrix.com_USB_Keyboard-event-kbd
t458765 (0x7000d)
e458761 (0x70009)
s458766 (0x7000e)
t458765 (0x7000d)
```

Which means : t → 0d ; e → 09 ; s → 0e ; t → 0d

- Encode de scancodes with modhex

```
netbook % modhex -h 0d090e0d
ctckcuct
```

- Configure the second slot of the yubikey

```
netbook % sudo ykpersonalize -2 -a00000000000000000000000000000000
  -ofixed=ctckcuct -o-strong-pw1 -o-strong-pw2 -o-static-ticket
  -oshort-ticket
```