GNUK TOKEN AND GNUPG SCDAEMON

“minimizing the attack surface”

NIIBE Yutaka <gniibe@fsij.org>

FOSDEM 2018

This is a talk of my experience

- to have better control (by its user)
- of computing for privacy
- with dedicate device
- of mimimized features

WHAT'S GNUK?

- Free Software Project under FSIJ
- Implementation of Cryptographic Token
- Supports OpenPGP card Protocol (v2 & v3)
- Runs on STM32103 MCU (Cortex-M3)
- Supports RSA-2048 and **ECC**

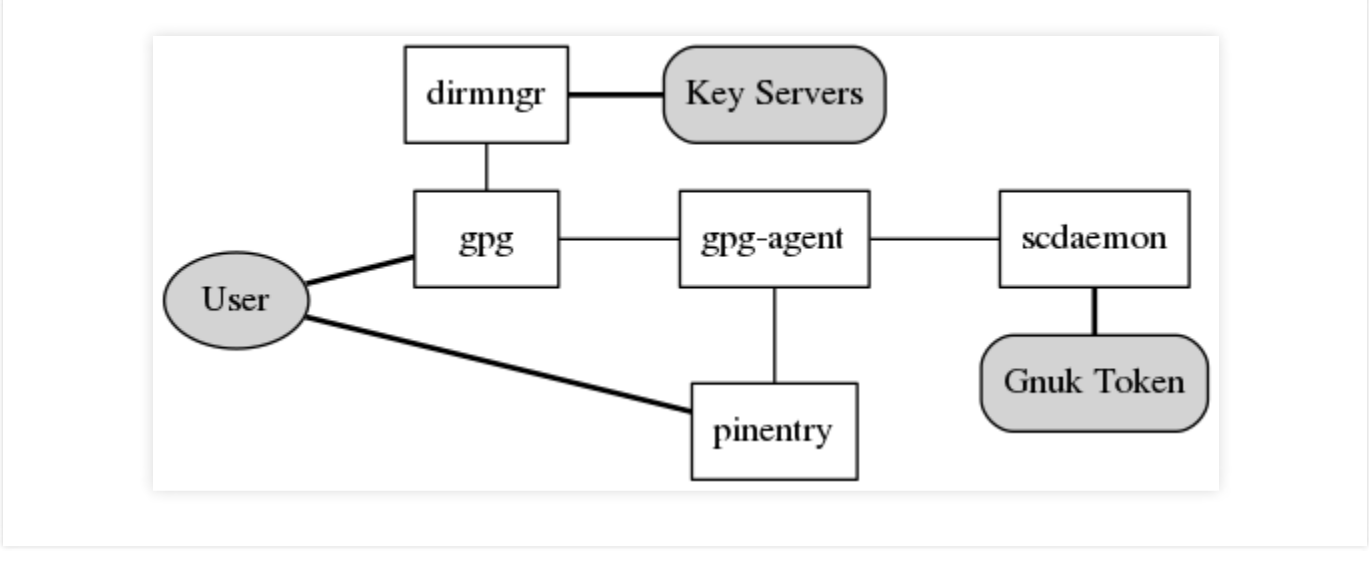
- Gnuk as *GNU* + *K* (*K* for _Key_)
- Gnuk as *G* + *NUK*

GNUK TOKEN

- Gnuk is software implementation
- We call a device “*Gnuk Token*” when it runs Gnuk

GNUK TOKEN AND SCDAEMON

By connected processes and a device



Reason: to minimize **the attack surface**

TYPICAL USE CASE

At work

Home

On the Go

THOUGHTS

- No more copy of private keys
- Separate dedicated device
- which is removable
- Supply power only on use
- Physically small surface

WHAT ARE LEARNED?

- Controlling my own computing: getting harder
- Random number sequence: No control by anyone
- Not only software toolchain, but also:
  - Tools like KiCAD, OpenOCD, sigrok
  - Firmware in JTAG device
  - Computer used in factory
- How to deliver the product

HARDWARE TARGET HISTORY

- Project started 2010 with Olimex
- STM32 part of STM8S Discovery Kit
- More boards support
- Reference hardware design in 2011
- Manufactured 1000pcs in 2012
- Update the design in 2016
- Manufactured 300pcs in 2017

SOON AFTER ITS START

Realized host side support is important

- CCID driver was typically for proprietary devices
- undocumented features, bad abstraction, no-good standardization (E.g.: pinpad support)
- Requirement for hardware deployment
- Joined GnuPG development in 2011 to improve `scdaemon`

SCDAEMON

Access smartcard through CCID reader

Difficult software, because of:

- Support for proprietary devices
  - Proprietary readers
  - Proprietary card
- Support for different OSes
  - GNU, *BSD, Windows, macOS, ...

SCDAEMON HAS BEEN IMPROVED

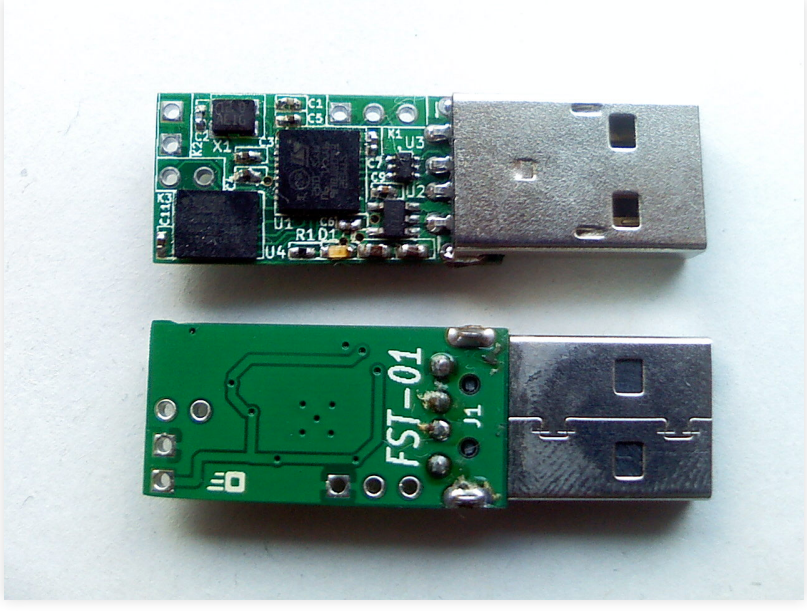
- Stable interaction between gpg-agent
- Robust access to device
- OpenPGP card v3 support
- No more PC/SC wrapper
- Direct access by libusb
  - Simultaneous use of multiple tokens

FST-01, FST-01G

FREE HARDWARE DESIGN

- For reproducible hardware implementation
- Simpler, no many features
- Use free tool: KiCAD

FST-01 DESIGNED IN 2011



FST-01G DESIGNED IN 2016

MANY OTHER THINGS

FLASHING MCU

- Reverse engineering for tool: ST-Link/V2

RANDOM NUMBER GENERATOR

NeuG started in 2011

- Entropy Source: 1/2-bit of each ADC sample
- No one should control (or can guess) instance of random number sequence

FIRMWARE UPDATE

This can be valid attack vector.

- Implemented in Gnuk and NeuG

USB VENDER ID

FSIJ got it in 2011 for Gnuk Project

MANUFACTURING

Free Hardware / Free Software friendly company

- Seeed Technology in ShenZhen
- They can distribute the product, too

RT OS

Chopstx started in 2013

GPL COMPLIANCE (1)

I tried with a serial ROM on FST-01

- to deliver source code on the device
- But, it takes time in production
- FAIL: manufacuring cost matters

GPL COMPLIANCE (2)

Fraucheky started in 2013

- Deliver GPL text on the device

DISTRIBUTION CHANNEL

- Seeed Studio (2012-2017) - w/ Gnuk 1.0.1
- Free Software Foundation (2015-) - w/ NeuG 1.0.x + SDcard of repo copy
- In person, at conferences
  - Debconf14, 15, 17
  - OpenPGP.conf in 2015

MANUFACTURING PROCESS IMPROVEMENT

Computer in factory matters

- BBG-SWD in 2016
  - SWD flashing tool by single board computer
  - to minimize the attack surface in factory

SOURCE CODE ACCESS

- By selling copy of repo of gniibe.org
- gitorious.org
- alioth.debian.org
- salsa.debian.org

USB EMULATION (SINCE 2017)

Support testing with no real hardware

ECO SYSTEM

- FSF: distribution of product
- GnuPG: gnuk-users mailing list
- Debian: source code repo
- FSIJ: USB Vendor ID (and travel cost)

HARDWARE SUGGESTIONS

- STM32 Nucleo F103 https://www.fsij.org/gnuk/neug-on-stm32-nucleo-f103.html
- Blue Pill http://wiki.stm32duino.com/index.php?title=Blue_Pill
- ST-Link/V2 clone
- FSF Shop https://shop.fsf.org/storage-devices/neug-usb-true-random-number-generator

SUMMARY (1)

Those things matter:

- Free Software on Host
- Free firmware on Device
- Free development environment
- Documented standard/protocol/interface
- Free tool
- Emulation for testing with no real hardware
- Distribution of product

SUMMARY (2)

Some dirty works/steps are required

- reverse engineering
- access by proprietary OS/tool/etc.
- business practice like USB VID
- bootstrap from proprietary env.

REFERENCES

- News: https://www.fsij.org/gnuk/
- Info: https://www.gniibe.org/category/fst-01.html
- Repo: https://salsa.debian.org/gnuk-team/

HAPPY HACKING!

“I want to free our people.
If you want to be free, join us.”
— Freysa to K