

# Hardware Backdooring is practical

Jonathan Brossard (Toucan System)



# DISCLAIMER

- We are not « terrorists ». We won't release our PoC backdoor.
- The x86 architecture is plagued by legacy. Governments know. The rest of the industry : not so much.
- There is a need to discuss the problems in order to find solutions...
- This is belived to be order of magnitudes better over existing backdoors/malware



# Agenda

- Motivation : state level backdooring ?
- Coreboot & x86 architecture
- State of the art in rootkitting, romkitting
- Introducing Rakshasa
- Epic evil remote carnal pwnage (of death)
- Why cryptography (Truecrypt/Bitlocker/TPM) won't save us...
- Backdooring like a state

# Who am I ?

- Security researcher, pentester
- First learned asm (~15 years ago)
- Presented at Blackhat/Defcon/CCC/HITB...
- Master in Engineering, master in Computer Sciences
- Co organiser of the Hackito Ergo Sum conference (Paris)

Likes : Unix, network, architecture, low level, finding 0days (mem corruptions).

Dislikes : web apps, canned exploits.

- Super pure English accent (French, learned English in India, lives in Australia... ;))

# FUD 101

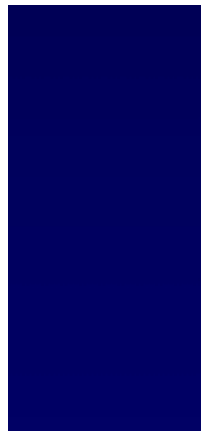


# Could a state (eg : China) backdoor all new computers on earth ?



Occupying the Information High Ground:  
*Chinese Capabilities for Computer Network Operations and Cyber Espionage*

This close relationship between some of China's—and the world's—largest telecommunications hardware manufacturers creates a potential vector for state sponsored or state directed penetrations of the supply chains for microelectronics supporting U.S. military, civilian government, and high value civilian industry such as defense and telecommunications, though no evidence for such a connection is publicly available.



Prepared for the U.S.-China Economic and Security Review Commission by Northrop Grumman Corp



Bryan Krekel  
Patton Adams  
George Bakos

March 7, 2012

**NORTHROP GRUMMAN**

# More introductory material

Cyberdéfense : les routeurs chi...

www.numerama.com/magazine/23225-cyberdefense-les-routeurs-chinois-accuses-d-etre-un-risque.html

## Cyberdéfense : les routeurs chinois accusés d'être un risque

Julien L. - publié le Jeudi 19 Juillet 2012 à 16h09 - posté dans [Société 2.0](#)

[Tweet](#) 37 [+1](#) 2

CC Partager [RSS](#)

Chine, Réseau, Windows, ZTE, Huawei 22 commentaires(s)

**Faudra-t-il se passer des équipements chinois dans le secteur des télécommunications ? Un rapport sénatorial dédié à la cyberdéfense avance cette idée, pointant du doigt les liens entre les industriels ZTE et Huawei et le pouvoir central chinois. Mais en matière de cyberdéfense, les matériels en provenance de l'Empire du Milieu ne sont pas les seuls à poser question.**

Les équipements de réseau chinois, un risque pour la cyberdéfense ? C'est ce qui ressort d'un rapport sénatorial conduit par Jean-Marie Bockel et [disponible sur le site](#) de la chambre haute du parlement. Si le document liste dix priorités et propose cinquante recommandations, l'une des pistes avancées par le sénateur socialiste a particulièrement surpris.

Cette priorité, la dixième, propose "d'interdire sur le territoire national et à l'échelle européenne le déploiement et l'utilisation de 'routeurs' ou d'autres équipements de cœur de réseaux qui présentent un risque pour la sécurité nationale, en particulier les 'routeurs' et certains équipements d'origine chinoise". Et deux sociétés sont directement citées dans le rapport : ZTE et Huawei

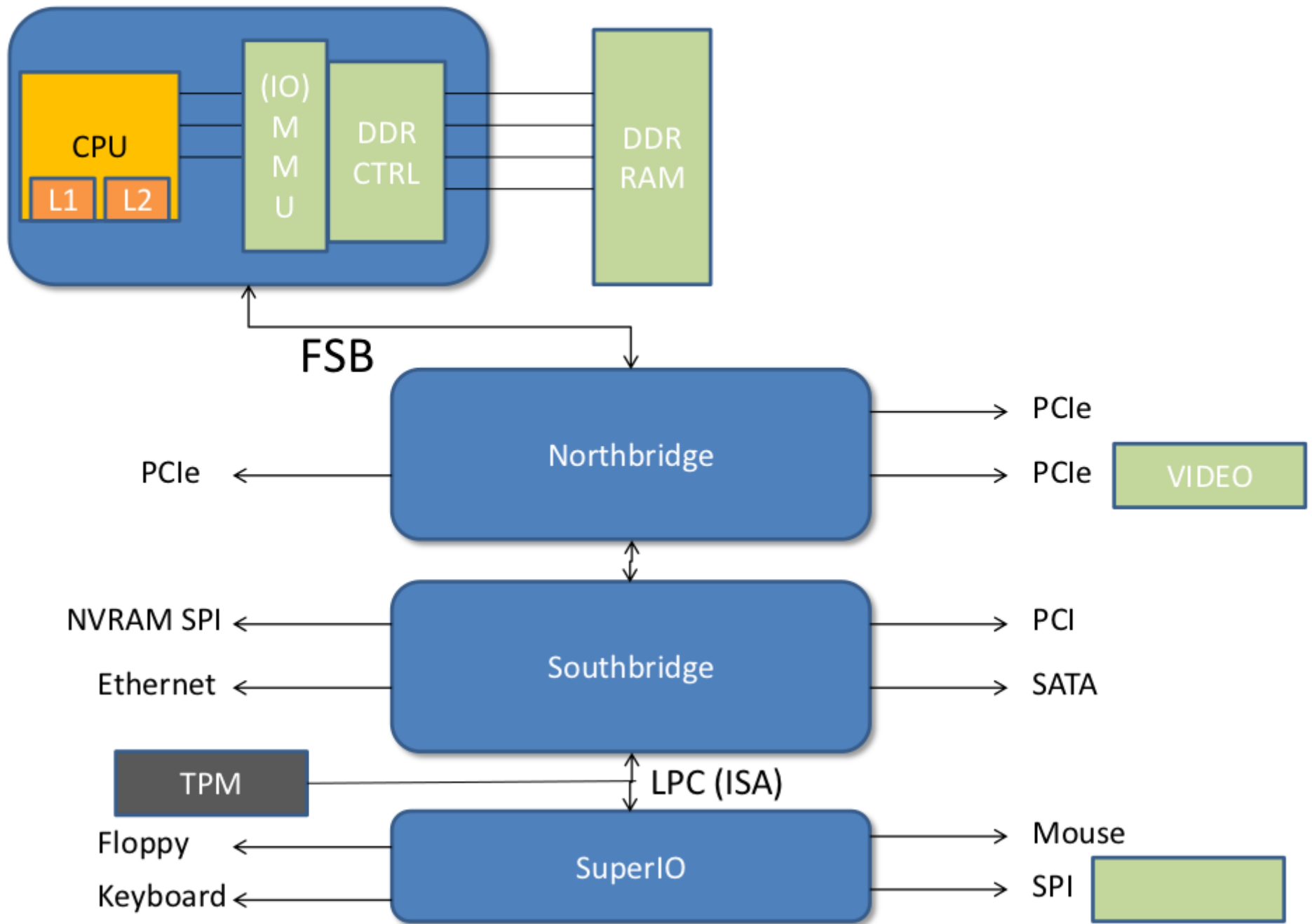


# Enough FUD...

## A bit of x86 architecture







# State of the art, previous work



# Previous work

- Early 80s : Brain virus, targets the MBR
- 80s, 90s : thousands of such viruses
- 2007, John Heasman (NGS Software) Blackhat US: backdoor EFI bootloader
- 2009, Anibal Saco and Alfredo Ortega (Core security), CanSecWest : patch/flash a Phoenix-Award Bios
- 2009, Kleissner, Blackhat US : Stoned bootkit. Bootkit Windows, Truecrypt. Load arbitrary unsigned kernel module.
- 2010, Kumar and Kumar (HITB Malaysia) : vbootkit bootkitting of Windows 7.
- Piotr Bania, Konboot : bootkit any Windows (32/64b)
- 2012 : Snare (Blackhat 2012) : UEFI rootkitting

# Introducing Rakshasa



# Goals : create the perfect backdoor

- Persistent
- Stealth (0 hostile code on the machine)
- Portable (OS independant)
- Remote access, remote updates
- State level quality : plausible deniability, non attribution
- Cross network perimeters (firewalls, auth proxy)
- Redundancy
- Non detectable by AV (goes without saying...)

# Rakshasa : Design (1/2)

- Core components :
  - Coreboot
  - SeaBios
  - iPXE
  - payloads

Built on top of free software : portability, non attribution, cheap dev (~4 weeks of work), really really really hard to detect as malicious.

- **Supports 230 motherboards.**

# Rakshasa : Design (2/2)

- Flash the BIOS (Coreboot + PCI roms such as iPXE)
- Flash the network card or any other PCI device (redundancy)
- Boot a payload over the network (bootkit)
- Boot a payload over wifi/wimax (breach the network perimeter, bypasses network detection, I(P|D)S )
- Remotely reflash the BIOS/network card if necessary

# Rakshasa : embedded features

- Remove NX bit → executable heap/stack.
- Make every mapping +W in ring0
- Remove CPU updates (microcodes)
- Remove anti-SMM protections → generic local root exploit
- Disable ASLR
- Bootkitting (modified Kon-boot payload\*)

\* Thanks to Piotr Bania for his contribution to Rakshasa :)



# Rakshasa : removing the NX bit (1/2)

MSR !!! Model Specific Register

AMD64 Architecture Programmer's manual (volume 2, Section 3.1.7 : Extended Feature Enable Register) :

*No-Execute Enable (NXE) Bit. Bit 11, read/write. Setting this bit to 1 enables the no-execute page-protection feature. The feature is disabled when this bit is cleared to 0.*

# Rakshasa : removing the NX bit (2/2)

; Disable NX bit (if supported)

```
mov    eax,0x80000000    ; get higher function supported by eax
cpuid    ; need amd K6 or better (anything >= 1997... should be ok)
```

```
cmp    eax,0x80000001
jb     not_supported    ; need at least function 0x80000001
```

```
mov    eax,0x80000001    ; get Processor Info and Feature Bits
cpuid
```

```
bt     edx,20            ; NX bit is supported ?
jnc    not_supported
```

```
movl   ecx, 0xc0000080    ; extended feature register (EFER)
rdmsr    ; read MSR
btr    eax, 11            ; disable NX (EFER_NX) // btr = bit test and reset
wrmsr    ; write MSR
```

not\_supported:

# Make every mapping +W in ring0

Intel Manuals (Volume 3A, Section 2.5):

*Write Protect (bit 16 of CR0) - When set, inhibits supervisor-level procedures from writing into read-only pages; when clear, allows supervisor-level procedures to write into read-only pages (regardless of the U/S bit setting; see Section 4.1.3 and Section 4.6). This flag facilitates implementation of the copy-on-write method of creating a new process (forking) used by operating systems such as UNIX.*

# Make every mapping +W in ring0 (32b/64b)

; 32b version :

```
mov eax,cr0
```

```
and eax,0xfffeffff
```

```
mov cr0,eax
```

; 64b version :

```
mov rax,cr0
```

```
and rax,0xfffeffff
```

```
mov cr0,rax
```

# Remove CPU updates (microcodes)

```
rm -rf ./coreboot/microcodes/
```

# Remove anti-SMM protections (1/2)

Intel® 82845G/82845GL/82845GV Graphics and Memory Controller datasheets, Section 3.5.1.22:  
SMRAM—System Management RAM Control Register (Device 0), bit 4 :

*SMM Space Locked (D\_LCK)—R/W, L. When D\_LCK is set to 1, D\_OPEN is reset to 0; D\_LCK, D\_OPEN, C\_BASE\_SEG, H\_SMRAM\_EN, TSEG\_SZ and TSEG\_EN become read only. D\_LCK can be set to 1 via a normal configuration space write but can only be cleared by a Full Reset. The combination of D\_LCK and D\_OPEN provide convenience with security. The BIOS can use the D\_OPEN function to initialize SMM space and then use D\_LCK to “lock down” SMM space in the future so that no application software (or BIOS itself) can violate the integrity of SMM space, even if the program has knowledge of the D\_OPEN function.*

# Remove anti-SMM protections (2/2)

D\_LCK is not supported by CoreBoot currently anyway...

; disable D\_LCK in Coreboot shellcode ;)

nop

# Rakshasa : embedded features : conclusion

- Permanent lowering of the security level on any OS.
- Welcome back to the security level of 1997.
- Persistent, even if HD or OS is remove/restored.



# Rakshasa : remote payload

- Bootkit future OSes
- Update/remove/reflash firmwares (PCI, BIOS)
- Currently capable of Bootkitting any version of Windows (32b/64b) thanks to special version of Kon-boot

# Rakshasa : stealthness

- We don't touch the disk. 0 evidence on the filesystem.
- The code flashed to motherboard is not hostile per se (there is one text file with urls in it.. that's it).
- We can remotely boot from an alternate payload or even OS : fake Truecrypt/Bitlocker prompt !
- Optionally boot from a WIFI/WMAX stack : 0 network evidence on the LAN.
- Fake BIOS menus if necessary. We use an embedded CMOS image. We can use the real CMOS nvram to store encryption keys/backdoor states between reboots.

# Rakshasa : why using Coreboot/SeaBios/iPXE is the good approach

- Portability : benefit from all the gory reverse engineering work already done !
- Awesome modularity : embed existing payloads (as floppy or cdrom images) and PCI roms directly in the main Coreboot rom !  
Eg : bruteforce bootloaders (Brossard, H2HC 2010), bootkits without modification.
- Network stacks : ip/udp/tcp, dns, http(s), tftp, ftp...  
make your own (tcp over dns? Over ntp ?)
- Code is legit : can't be flagged as malware !

# DEMO : Evil remote carnal pwnage (of death)



**I can write blogs too... Muhahahaha...**

# How to properly build a botnet ?

- HTTPS + asymmetric cryptography (client side certificates, signed updates)
- Fastflux and/or precomputed IP addresses

If Microsoft can do secure remote updates, so can a malware !

- Avoid DNS take overs by law enforcement agencies by directing the C&C rotatively on innocent web sites (are you gonna shut down Google.com?), use asymmetric crypto to push updates.

# Why crypto won't save you...



# Why crypto won't save you (1/2)

- We can fake the booting/password prompt by booting a remote OS (Truecrypt/Bitlocker)
- Once we know the password, the BIOS backdoor can emulate keyboard typing in 16b real mode by programming the keyboard/motherboard PIC microcontrollers (Brossard, Defcon 2008)
- If necessary, patch back original BIOS/firmwares remotely.

# Why crypto won't save you (2/2)

TPM + full disk encryption won't save you either :

1) It's a passive chip : if the backdoor doesn't want explicit access to data on the HD, it can simply ignore TPM.

2) Your HD is never encrypted when delivered to you. You seal the TPM when you encrypt your HD only. So TPM doesn't prevent backdooring from anyone in the supply chain.



# How about Avs ??

- Putting an AV on a server to protect against unknown threats is purely cosmetic.
- You may as well put lipstick on your servers...



# Example : 3 years old bootkit



SHA256: 214ce3ce21e38ea145ba2cd52cce7e94367a2701ea5f4efda4a1cc248fbec1d2

File name: konFLOPPY.img

Detection ratio: 2 / 43

Analysis date: 2012-03-07 07:14:43 UTC ( 3 weeks, 3 days ago )

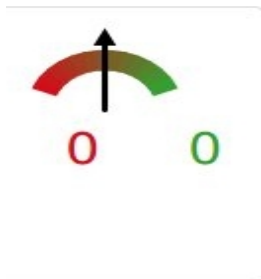


Kaspersky	-	20120307
McAfee	-	20120307
McAfee-GW-Edition	Heuristic.BehavesLike.Exploit.CodeExec.EPMG	20120307
Microsoft	-	20120307
NOD32	-	20120307
Norman	norn virus, B.H	20120304
nProtect	-	20120306

# Example : 3 years old bootkit (+ simple packer)



SHA256: 8:  
File name: k.  
Detection ratio: 0  
Analysis date: 21



- Antivirus
- AhnLab-V3
- AntiVir
- Antiy-AVL
- Avast
- AVG
- BitDefender
- ByteHero

CAT-QuickHeal	-	20120331
ClamAV	-	20120331
CommTouch	-	20120330
Comodo	-	20120331
DrWeb	-	20120331
Emsisoft	-	20120331

# Realistic attack scenarii



# Realistic attack scenarii

- Physical access :

Anybody in the supply chain can backdoor your hardware. Period.

Flash from a bootable USB stick (< 3mins).

- Remote root compromise :

```
If (OS == Linux) {
```

```
    flash_bios;
```

```
} else {
```

```
Pivot_over_the_MBR ;
```

```
}
```

# Realistic attack scenari

D-link DGE 530T dge530t 1000MT Gigabit Desktop PCI NETWORK NIC CARD 10/100/1000 | eBay - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Index of file:///opt/rakshasa/d... Low Cost Embedded x86 Tea... Index of file:///opt/rakshasa/d... Pinczakko's Guide to Self-pa... D-link DGE 530T dge530t 10...

www.ebay.com/itm/D-link-DGE-530T-dge530t-1000MT-Gigabit-Desktop-PCI-NETWORK-NIC-CARD-10-100-1000-/140706820838?pt=LH\_DefaultDomain\_0&hash=item20c2c7b2e6

Most Visited Tasks Ralf Brown HES 2012 HES orga My box Linux/i386 system c... Reverse IP Lookup ... http://www.zonabat... http://www.mgid.co... The Art of Assembly... DEF CON® 19 Hack... Jeu d'instruction x86

ebay® Welcome! Sign in or register.

CATEGORIES ELECTRONICS FASHION MOTORS TICKETS DEALS CLASSIFIEDS eBay Buyer Protection Learn more

Back to search results Computers & Networking > Networking & Communications > Network Interface Cards > PCI Network Cards > Gigabit & Up

This is a private listing. Sign in to view your status or learn more about private listings.

**FREE shipping**

**D-link DGE 530T dge530t 1000MT Gigabit Desktop PCI NETWORK NIC CARD 10/100/1000**

Item condition: **Used**

Quantity:  More than 10 available / 20 sold

US \$8.94 **Buy It Now** **Add to cart** **Add to Watch list**

**NEW! eBay shopping cart**  
Shop, compare and buy several items at once with your shopping cart.

Share: **Add to Watch list**

**Top-rated seller**  
so great.value (23015)   
98% Positive feedback  
Consistently receives highest buyers' ratings  
Ships items quickly  
has earned a track record of excellent service

Save this seller  
See other items  
Visit store: Value.to.Buy

**Bill Me Later** \$10 back on 1st purchase & 6 months to pay  
Subject to credit approval. [See terms](#)

Shipping: **FREE** - ePacket delivery from China | [See all details](#)  
See details about international shipping here.   
Item location: **Guangzhou, China**  
Ships to: **Worldwide** [See exclusions](#)

Delivery: Estimated between **Sat. Apr. 21** and **Fri. Apr. 27**

Payments: **PayPal**, **Bill Me Later** | [See details](#)

Returns: 14 days money back, buyer pays return shipping | [Read details](#)

**eBay Buyer Protection**  
Covers your purchase price plus original shipping.  
[Learn more](#)

**Buy and sell on the go!**  
It's easy with the eBay Mobile app  
**Download FREE app**

Ad Feedback | AdChoice

**Description** **Shipping and payments** [Print](#) | [Report item](#)

Seller assumes all responsibility for this listing. Item number: 140706820838

**Item specifics**

Condition: Used: An item that has been used previously. The item may have some signs of wear. Brand: D-link

# BONUS : Backdooring the datacenter





## Using iPXE in VMware

You can replace the default VMware PXE ROM with an iPXE ROM, which will enable you to boot your virtual machine via HTTP, iSCSI, NFS, or any other protocol supported by iPXE.

### Table of Contents

- Selecting the network adapter
- Building the ROM images
- Configuring the virtual machine
- Booting the virtual machine

### Selecting the network adapter

VMware is capable of installing several network adapters:

VMware name	iPXE driver name	PCI vendor/device IDs	iPXE ROM image
e1000	e1000	8086:100f	8086100f.rom
e1000e	e1000e	8086:10d3	808610d3.rom
vmxnet	pxnet32	1302:7088	13027088.rom
vmxnet1	(not supported)	1302:0729	
vmxnet3	vmxnet3	1302:0790	13020790.rom



Select one of the supported network adapters, and ensure that your virtual machine is configured to use this adapter. You can do this by editing the `.vmx` file that defines your virtual machine, and changing the setting

```
ethernet0.virtualDev = "e1000"
```

For example, to select an e1000 network adapter:

```
ethernet0.virtualDev = "e1000"
```

### Building the ROM images

Download iPXE and then build ROM images for all of the supported network adapters using:

```
wake bin/8086100f.rom bin/808610d3.rom bin/13027088.rom bin/13020790.rom
```

Copy the iPXE ROM images `8086100f.rom`, `808610d3.rom`, `13027088.rom` and `13020790.rom` to a suitable location (e.g. to the directory `/usr/lib/ovmf/resources/`).

### Configuring the virtual machine

Edit the `.vmx` file that defines your virtual machine, and add the following lines:

```
ethernet0.address0 = 0x01:00:00:00:00:00  
ethernet0.firmware = "/usr/lib/ovmf/resources/8086100f.rom"  
ethernet0.firmware = "/usr/lib/ovmf/resources/808610d3.rom"  
ethernet0.firmware = "/usr/lib/ovmf/resources/13027088.rom"  
ethernet0.firmware = ""  
ethernet0.firmware = "/usr/lib/ovmf/resources/13020790.rom"
```

(Replacing `/usr/lib/ovmf/resources/` with the name of the directory to which you copied the iPXE ROM images).

### Booting the virtual machine

Boot your virtual machine in the usual way. You should see VMware detect and use the iPXE ROM:

```
iPXE (001:001:0000:00:00:00:00:00) vmlinuz=linux-ovmf-efi-64
```



# Remediation



# Remediation (leads)

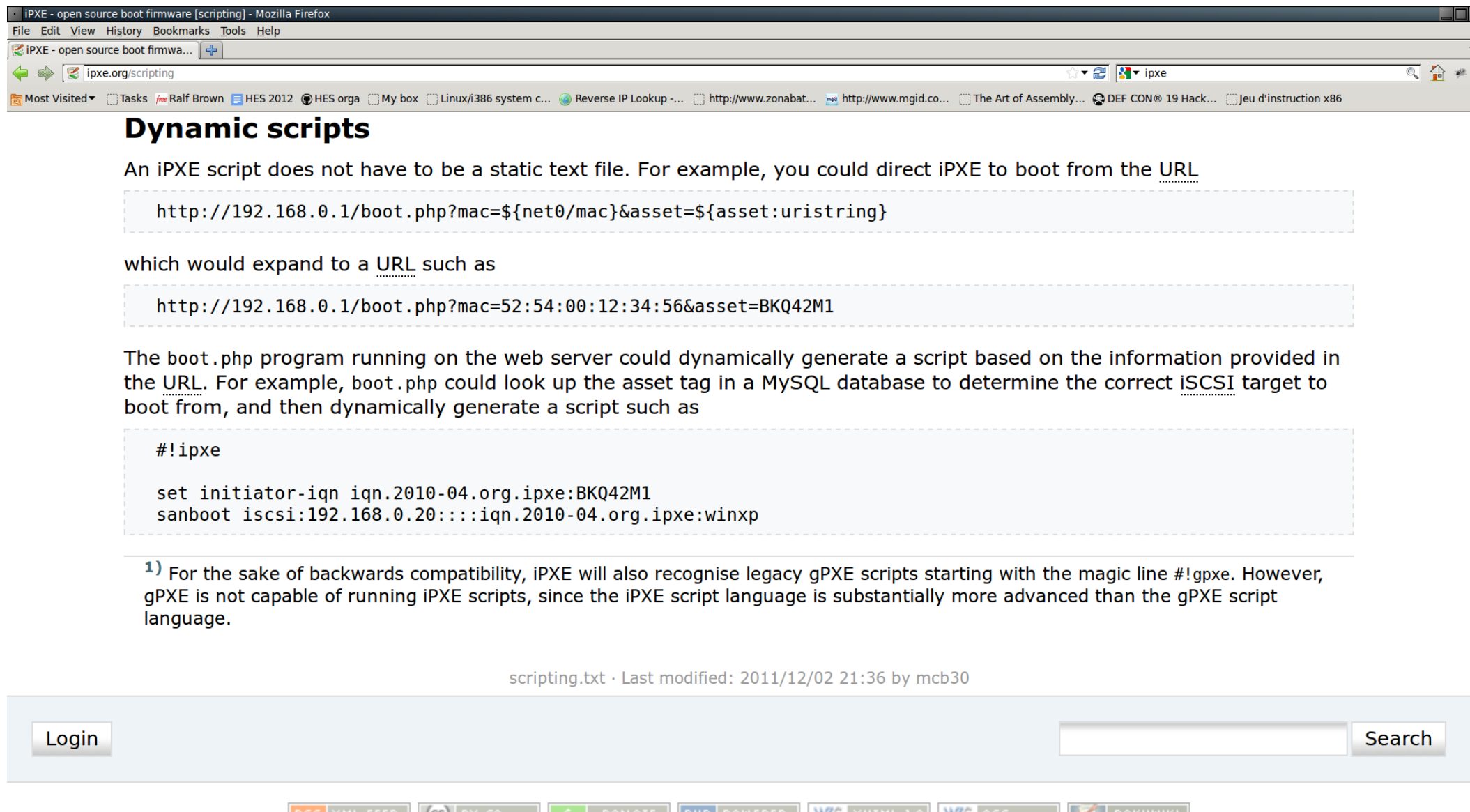
- Flash any firmware upon reception of new hardware with open source software you can verify.
- Perform checksums of all firmwares by physically extracting them (FPGA..) : costly !
- Verify the integrity of all firmwares from time to time
- Update forensics best practices :
  - 1) Include firmwares in SoW
  - 2) Throw away your computer in case of intrusion

Even then... not entirely satisfying : the backdoor can flash the original firmwares back remotely.

# Side note on remote flashing

- BIOS flashing isn't a problem : the flasher (Linux based) is universal.
- PCI roms flashing is more of a problem : flasher is vendor dependant...

# Detecting network card manufacturer from the remote C&C



**Dynamic scripts**

An iPXE script does not have to be a static text file. For example, you could direct iPXE to boot from the URL

```
http://192.168.0.1/boot.php?mac=${net0/mac}&asset=${asset:uristring}
```

which would expand to a URL such as

```
http://192.168.0.1/boot.php?mac=52:54:00:12:34:56&asset=BKQ42M1
```

The boot.php program running on the web server could dynamically generate a script based on the information provided in the URL. For example, boot.php could look up the asset tag in a MySQL database to determine the correct iSCSI target to boot from, and then dynamically generate a script such as

```
#!ipxe
set initiator-iqn iqn.2010-04.org.ipxe:BKQ42M1
sanboot iscsi:192.168.0.20:::iqn.2010-04.org.ipxe:winxp
```

1) For the sake of backwards compatibility, iPXE will also recognise legacy gPXE scripts starting with the magic line `#!gpxe`. However, gPXE is not capable of running iPXE scripts, since the iPXE script language is substantially more advanced than the gPXE script language.

scripting.txt · Last modified: 2011/12/02 21:36 by mcb30

Login  Search

# Backdooring like ~~NSA~~ China



# Backdooring like a state

## Rule #1 : **non attribution**

- you didn't write the free software in first place.
- add a few misleading strings, eg : in mandarin ;)

## Rule #2 : **plausible deniability**

- use a bootstrap known remote vulnerability in a network card firmware (eg : Dufлот's CVE-2010-0104)
  - « **honest mistake** » if discovered.
- remotely flash the BIOS.
- do your evil thing.
- restore the BIOS remotely.

# More DEMOS



# Outro

This is not a vulnerability :

- it is sheer bad design due to legacy.
- don't expect a patch.
- fixing those issues will probably require breaking backward compatibility with most standards (PCI, PCIe, TPM).



# Questions ?

