Hit by a Bus:

Physical Access Attacks with Firewire

Presented By Adam Boileau

Ruxcon 2006

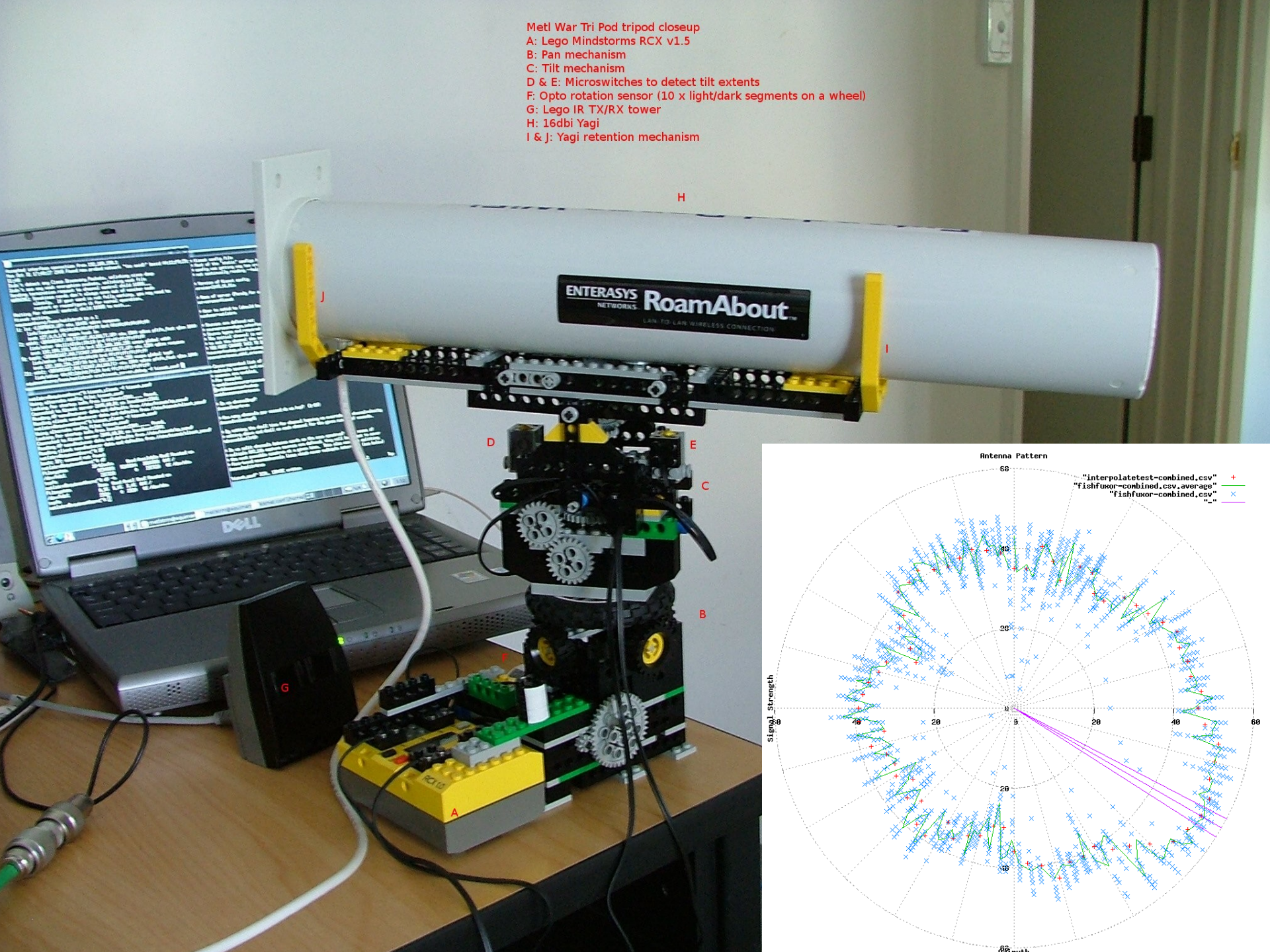
security-assessment.com

About Me

- Adam Boileau, Senior Security Consultant, Security-Assessment.com
- There's a few of us here this year...
- I'm a Unixy, networky sorta guy, I like python, I play with wireless.
- You might remember me from Rux last year...
- ... or maybe you've seen my robotic lego yagi scoping out your wireless networks :)


Metl War Tri Pod Components
A: Metl War Tri Pod, lego mindstorms, brickOS
B: 16dbi Yagi
C: Lego IR TX/RX Tower
D: Zcom 300mW PCMCIA card
E: Kismet
F: MWTP-Kismet client
G: Visualisation bit of MWTP-Kismet
H: LNP Lego Networking daemon
I: MWTP Tripod Telemetry daemon
J: MWTP Logging daemon
K: MWTP Remote Controller interface
L: USB Gamepad, remote tripod controller
M: Moses, token blasphemy
N: Gin & Tonic
HITACHI
YELLOW PAGES
YELLOW PAGES

Metl War Tri Pod tripod closeup
A: Lego Mindstorms RCX v1.5
B: Pan mechanism
C: Tilt mechanism
D & E: Microswitches to detect tilt extents
F: Opto rotation sensor (10 x light/dark segments on a wheel)
G: Lego IR TX/RX tower
H: 16dbi Yagi
I & J: Yagi retention mechanism



In Which I Say What I'm Going To Say

- Intro (<-- you are here)
- Physical Access Attacks
- The Fire! in your Wire
- '-._.-'-._.-Demo-._.-'-._.-'-
- (Ab)uses of Firewire
- '-._.-'-._.-Demo-._.-'-._.-'-
- Mitigation
- You Will Need...
- Q&A

Physical Access Attacks

- **Assertion: Physical access to a general purpose computer is game over**
- **Qualifiers: “general purpose”**

  Systems that are not designed to operate in a hostile environment. Commodity kit.

  Compare with ATMs, custom kiosks, parking meters
- **Traditional physical access attacks include:**

  Booting off disk/CD/USB to gain raw disk access

  Bypass BIOS password with the CMOS reset jumper

  Hardware keyboard loggers
- **Hardened systems also get attacked:**

  Side channel attacks on crypto smartcards (Ruxcon 03!)

  ATM data circuit MitM

- **ATMs: hardened hardware for hostile environments!**
- **But I sure hope their crypto is turned on...**

Traditional Attacks & Mitigation

Attack	Countermeasure
• Boot off disk/cd	• BIOS Password
• Open case, jump BIOS	• Case locks, intrusion switches
• Steal disk/machine	• Kensington cables, disk cryptography
• Keyboard loggers	• Physical inspection, two factor auth, biometrics

In general, attacks are not at all stealthy, and the majority of environments defend by "having someone keep an eye on things".

Physical Access Attacks

- **Increasingly, commodity kit is being used where specialised used to be**
- **Public access systems are more common:**
  - net cafes
  - internet kiosks
  - self service systems

Physical Access in 2006

- Walk in to a Net cafe, or up to a kiosk
- Plug in your Firewire iPod.
- Is anyone going to look twice?
- The guy who's keeping an eye on things wont.

- Embedded, handheld devices like ipods, cellphones etc. are becoming more and more capable, and like “real” computers.
- Physical access attacks are now much more subtle than before.
- And you can't protect against them because “physical access wins”
- Even Microsoft says so.

Firewire Background

- Firewire (IEEE-1394) is a peripheral connection bus standard
- Developed early 90s by Apple, TI, Sony (who call it “iLink”)
- A serial bus, data rates 100 - 800mbps, over copper or fibre
- Present on most laptops, high-end desktops
- Loosing in popularity to USB-2 for most things.
- Popular in the broadcast industry for it's support for isochronous transfers (guaranteed bandwidth)
- For most people, is just “Betamax USB”

Firewire vs USB

- **USB is a serial bus for low-speed, external peripherals; printers, scanners, mice, keyboards**

  Sure it does 400mbps now, but only cause Intel got sick of the Firewire politics
- **Firewire is a high-speed serial bus, designed for bus-mastering, isochronous data exchange for real-time applications**
- **Compare these two lists:**

- **Expansion Bus**

  PCI/AGP

  ISA

  PCMCIA/Cardbus

  Firewire!

- **Peripheral Bus**

  Serial Port

  Parallel Port

  PS2/AT

  USB

The Fire in your Wire

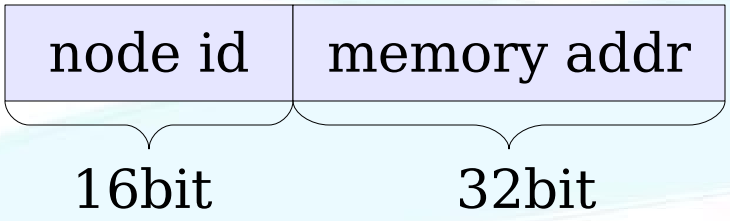
- **Peripheral busses have**
  A client-server model
  A wire protocol that's abstracted from the implementation
  Devices are restricted to what the protocol defines

- **Expansion busses have**
  A peer-to-peer bus-mastering model
  Direct bus (and therefore memory) access
  Limited only by the creativity of the device engineer

- **In terms of how you use them, they're the same**
- **But in terms of how they work, USB is the former and Firewire is the latter**

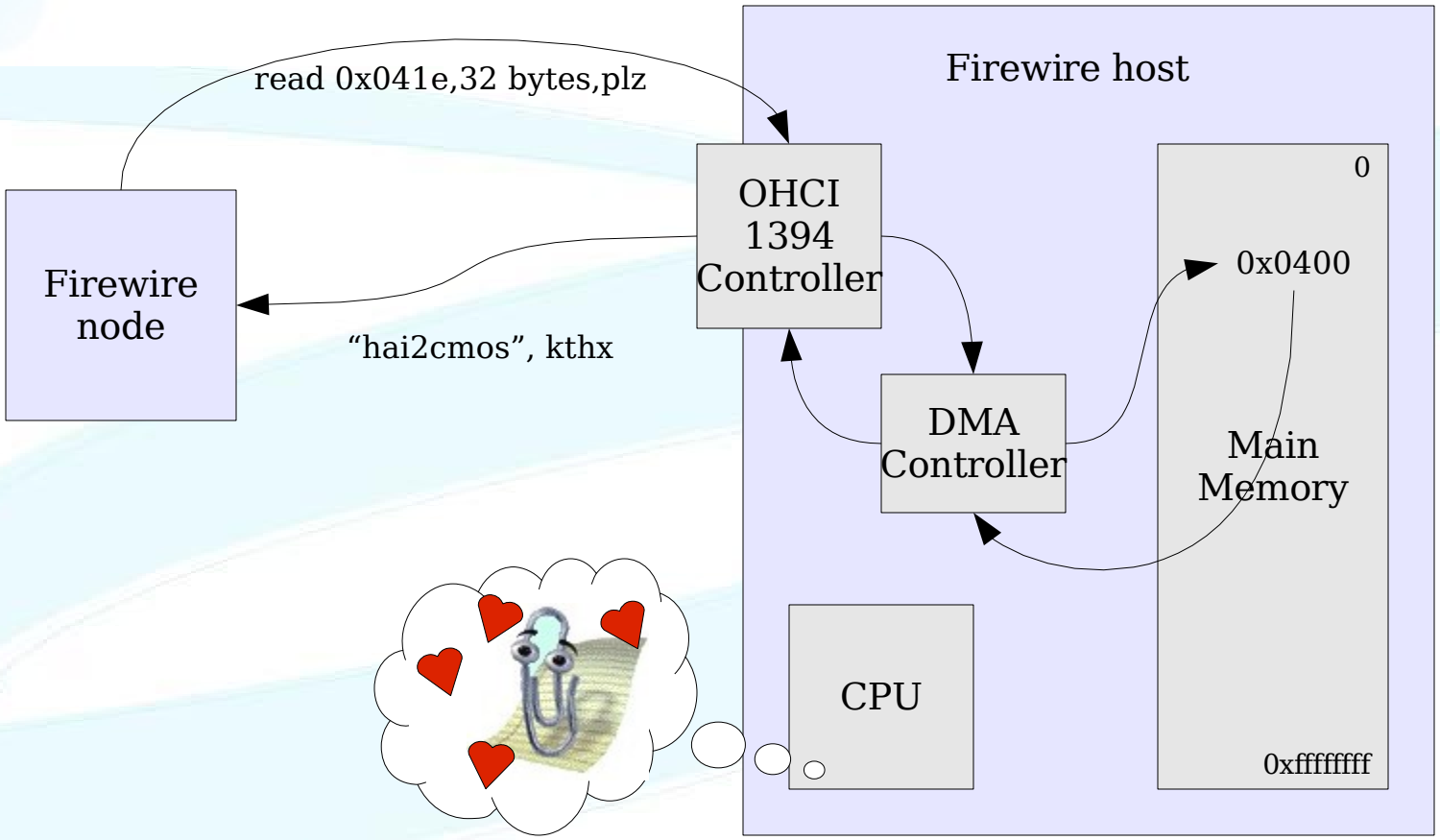
Firewire Addressing

- **Nodes on a firewire bus address each other like this**



- **Yes, that is a memory address...**
- **... and yes, it's 32 bits long**
- **and yes, it does map onto the bottom 4GB of physical memory**

DMA: The Fire in your Wire



In a nutshell

- With Firewire, I can read and write main memory...
- ...without the OS being involved...
- ...because that's how it's meant to work.
- (just like I could if I were to plug a PCMCIA card in... which is what h1kari presented at Shmoocon 2k6, apparently)

This aint 0day

- “Quinn The Eskimo” won best Mac Hack 2003 for a remote Firewire screensaver
- First security discussion by Max Dornseif at PacSecJP 2004: “Owned by an iPod”, then again at CanSecWest 2005

- So why am I up here talking about it?
- It's interesting because it's a feature, not a bug.
- Compare with Maynor's USB and WLAN driver bugs
- Many people still haven't heard about it
- Oh, and one other thing...
- ... all the previous public discussion says this:

	read	write
MacOS	works	works
FreeBSD	works	works
Linux	nope	works
Windows 2000	CRASH	CRASH
Windows XP	nope	nope

Demo

A vict^h^h^holunteer with a Win XPSP2 laptop with Firewire...

So, obviously

- it works just fine against Windows.
- There's some special sauce, courtesy of my colleague TMASKY </applause>

- You need to tell Windows that you're a device who deserves DMA access...
- ...by lying about your configuration.
- Firewire configuration is done by a reserved bit of memory address space called the Config Status Register
- Each node requests and parses other nodes CSRs from address `0xffff0000000`
- This describes the capabilities of the device

CSR Trickery

```
root@host~# romtool
Usage:
        Set the Firewire CSR for a port: romtool -s port romimagefile
        Get the Firewire CSR for a port: romtool -g port romimagefile
               Snarf another node's CSR: romtool -o port node romimagefile

root@host~# echo “My ipod is plugged in, and is port 0, node 0”
My ipod is plugged in, and is port 0, node 0

root@host~# romtool -o 0 0 omgipod.csr
Wrote 1024 byte ROM image of device on port: 0 node: 0 to omgipod.csr

root@host~# romtool -s 0 omgipod.csr
Updated 1024 byte ROM image from omgipod.cs

root@host~# echo “zomg, now I have two ipods!”
zomg, now I have two ipods!
```

(Ab)uses of Firewire

- **Bona-fide**

  Forensic memory imaging

  Remote debugging
- **50/50**

  Recovering passwords, crypto keys, etc. from memory

  Pulling video memory (to get the contents left over from other video modes)
- **Downright nasty**

  Bypassing authentication

  Owning stuff (e.g. escalating privs, dropping trojans)

  Anything you want. You have read/write to memory!

Memory Forensics

- **Traditional hardware devices very expensive**
- **Firewire is easy, hot-pluggable, and cheap**
- **Turn up to an incident, image memory of the system before you image the disks**

  Catch memory-based stealth rootkits/trojans

  All the kernel structures for open files, sockets, processes

  Capture in memory images of programs running, e.g. exploit tools, encrypted/packed stuff
- **Downsides:**

  Tools for reconstructing memory images into useful data are immature

  It's not guaranteed reliable; you can crash it if you're not careful about certain blocks of memory

Memory Forensics, Just Like That

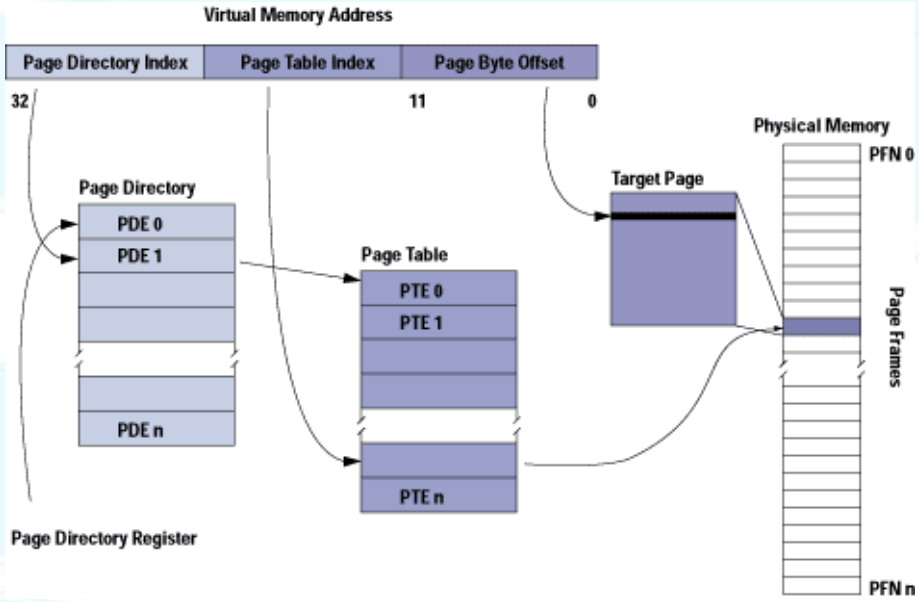
```
root@host~# 1394memimage 0 0 test1 -50M
1394memimage v1.0 Adam Boileau, 2006. <adam@storm.net.nz>
Init firewire, port 0 node 0
Reading 0x03135000 (50388KiB) at 3147 KiB/s...
52428800 bytes read
Elapsed time 16.27 seconds
Writing metadata and hashes...

root@host~# ls test1*
test1     test1.md5        test1.sha        test1.meta

root@host~# cat test1.meta
Forensic Firewire Memory Image Metadata
Using 1394memimage v1.0 Adam Boileau, 2006. <adam@storm.net.nz>
Memory range: 0x00000000-0x03200000 (52428800 bytes)
Started: Tue Sep 19 19:25:21 2006
...
```

Down & Dirty with Physical RAM

- Read / write to physical RAM is great, but...
- It's not all easy.
- Physical RAM pages (typically 4KiB) are three levels of indirection away from what userland sees (a virtual address)



Down & Dirty with Physical RAM

- **Physical RAM is fragmented, fast changing, and you never know who's caching what, where**
- **If you can find the kernel page table structures, you could work backwards, but very OS dependent**
- **For example, in winlockpwn, I have to find MSGina.DLL in memory:**

  I know a signature, and it's offset from the beginning of the page when loaded into memory

  I know it's not going to be in kernel memory, so I skip the bottom hundred or so megs

  I check each page for the signature at a fixed offset

  Approx 20x faster than just reading all memory and pattern matching. (length of sig vs 4KiB)

Password recovery

- **Passwords and key material live in RAM**
- **Real mode keyboard interrupt buffer in the BIOS Data Area contains the last 16 bytes typed before you went to protected mode...**

  Such as your PGP Wholedisk Passphrase

  or your BIOS boot password
- **Everything else that's running (IM clients, browsers, password-storage apps, OS etc)**
- **So, before you walk off with a laptop, plug in, image it's memory, now you can decrypt the disk at your leisure, or use the passwords as cribs for cracking other stuff**

Realmode Password Recovery

Demo

Speculation & Tomfoolery

- **You can do pretty much anything via Firewire**
- **You could:**

  Implement a remote keylogger

  Blit an image onto the screen (remote screensaver)

  Change the UID of a process to zero (ala Max)

  Write a DLL injector and drop rootkits

  Write a software firewire disk emulator (Hi Oracle!)

  Perform crossview memory-stealth-rootkit detection (ddefy-defy!)
- **In other words, what ever you're crazy enough to dream up, and cunning enough to implement**
- **Firewire is great! You should make sure all your machines have Firewire! >:D**

Mitigation

- **The OHCI spec doesn't provide much scope for mitigation. :(**
- **Turn off your Firewire port in hardware**
- **On G5 Macs, enable an Open Firmware password**
- **Disable the OS OHCI controller drivers**
- **In Linux, modprobe ohci1394 phys_dma=0**
- **Fill your Firewire ports with epoxy**
- **Some software (usually endpoint security solutions) provide tools to control USB and Firewire. Some work, some don't.**

  In general, ones that restrict individual devices (eg “no ipods”) won't work.

You will need

- **A firewire capable linux box**
- **Linux kernel with firewire support, and raw1394**
- **read/write to /dev/raw1394**
- **My pythonraw1394 bindings**
  requires python and libraw1394 to run
  SWIG and libraw1394-dev to build
- **Optionally a CSR image of a storage or similar device**
  Only if you're targeting Windows systems
- **A tool to do whatever you want to the target**
  1394memimage, winlockpwn, fireversion

High Level Python Raw1394 Bindings

```
root@host~# cat businfo
#!/usr/bin/python
# Python raw1394 test
# Metlstorm 2k6

import sys
import struct
import firewire
h=firewire.Host()
print "Firewire initialized, with %d ports available:" % (len(h.ports))
print "Enumerating port & node tree..."
for p in h:
        print p
        for n in p:
                print n
                print n.getConfigROM()
```

Other Lunacy

- **From the 1394 Trade Association Website:**

  "Dallas, December 8, 2003 – The 1394 Trade Association's Wireless Working Group today announced that the specification for Wireless 1394 applications is functionally complete and ready for a ballot as early as January 2004."
- **Yep, Firewire over wireless.**
- **Targeting layer 3, over 802.11n, 802.15.3 or some other UWB PHY.**
- **Anyone else think this sounds fun? Who needs Maynor/JohnyCache technique now? ;)**
- **Another press release:**

  "Dallas, September 6, 2006 - Czech Republic's National Court System is Largest Single-Site IEEE 1394 (FireWire) Installation"

Props

- Not just me; an SA team effort
- Tmasky for the Windows CSR idea
- Antic0de for the windows shellcode technique
- Darren for pimping me as “mitigation”

Links & References

- **My projects (including the tools from this presentation) & other bollocks:**

  http://www.storm.net.nz
- **Max Dornseif, original Firewire security presentations and others:**

  http://md.hudora.de/
- **Discussion of crashes while imaging the upper memory area via firewire:**

  http://ntsecurity.nu/onmymind/2006/2006-09-02.html
- **Microsoft's 10 Immutable Laws of Security**

  http://www.microsoft.com/technet/archive/community/columns/security/essays/10imlaws.mspx
- **H1kari's Shmoocon 2006 “Cardbus Bus Mastering”**

  Can't find a link :(

Questions ?

http://www.security-assessment.com

adam.boileau@security-assessment.com