Embedded Systems
–
"Invisible" Devious Devices?

Sergey Bratus

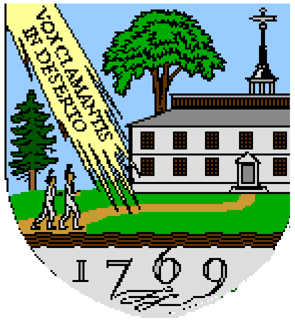
sergey@cs.dartmouth.edu

ERNW Living Security.

DARTMOUTH 1769

The speaker

- **Sergey Bratus**
- **Research Assistant Professor, Dartmouth College, USA**



Agenda

- Definition
- Spot the embedded system
- Embedded systems then and now
- Dan Geer's perspective: embedded systems as life forms
- How to hack them
- „War stories"
- What can we do? The engineering challenges.
- Conclusions & Outlook

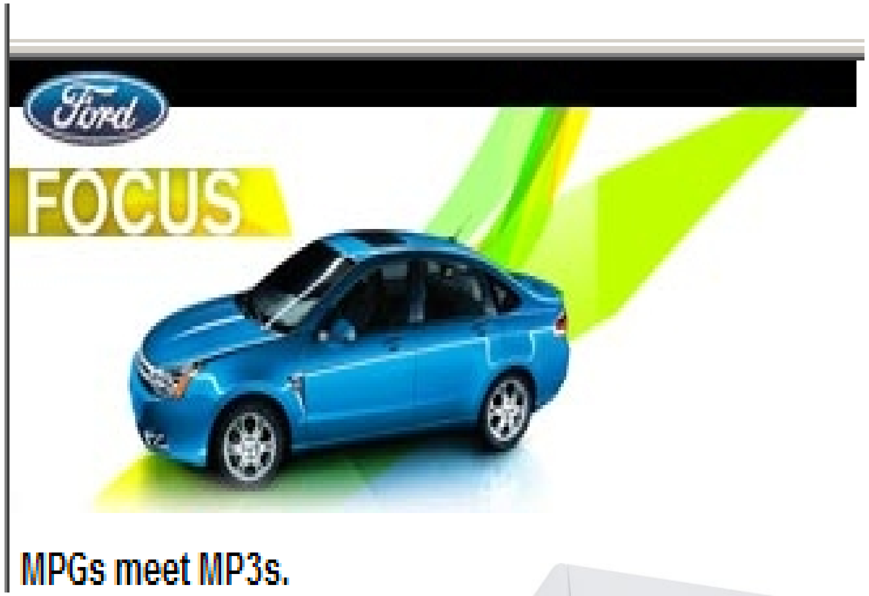
Embedded Systems – Definition

WIKIPEDIA
The Free Encyclopedia

tells us: “An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions“

However, this definition is misleading.

Spot the embedded system

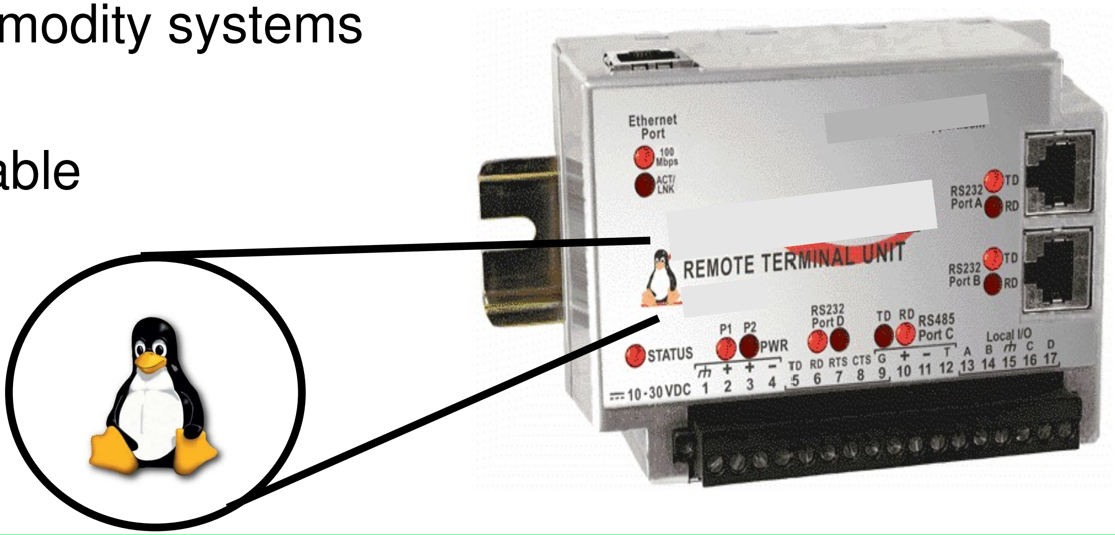


Intuitions: then...

- **Embedded devices are everywhere**
  - Multimedia, kiosks, phones
  - Cars
  - SCADA systems (e.g., power grid)
- **<u>Used to be</u> underpowered, highly specialized platforms**
  - No “slack” for doing it anything except their normal work
  - <u>*“If it works as expected, it could not possibly be doing anything else”*</u>
  - Only got looked at when not working right
  - Expensive SDKs, not available to general public
  - Not connected to Ethernet networks (e.g., use serial/RS232, CAN bus)
- **<u>Used to be</u> expensive to exploit, low on the list of threats**

Intuitions: ...and now

- **Now tend to use commodity OSes**
  - Linux (e.g., MontaVista), Windows 2K/XP (including XP Pro)
  - Platforms are powerful enough to accommodate ***generic OS kernels***, libraries, HTTP servers for configuration access, ...
    - Boards not designed for particular narrow applications
    - Kernels support full TCP/IP networks stacks, many protocols
  - Programming for commodity systems is cheaper and faster
  - SDKs are freely available cross-compilation kits
  - So, how often are they **patched**?



A non-trivial problem

- **Commodity software => commodity problems :-)**

"*Some reports, such as the case of the Conficker outbreak within Sheffield Hospital's* ***operating ward****, suggest that even security-conscious environments may elect to* ***forgo automated software patching****, choosing to trade off vulnerability exposure for some perceived notion of* ***platform stability****...*"

– **http://mtc.sri.com/Conficker/**

“Evolutionary” insights (1)

- **Dan Geer on embedded systems:** *(SourceBoston 2009)*

“Lest you think that it is too far fetched to consider a computer a **life form**, subject to **evolution** just like any other life form, consider embedded systems. They are already **two orders of magnitude** more numerous than keyboards and displays... Hence **the future threat space ...** is a threat space **where a computer is not identifiable as such**, but is instead inside some nondescript appliance.”

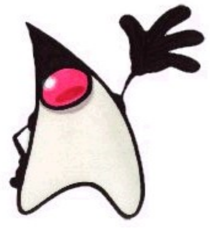
- Without a remote management interface and out of reach -- destined to **die** to make way for new generations
  - No way to fix late-discovered flaws, too many deployed systems
- With a remote management interface, must be **self-protecting**
  - Or else skilled attackers not only take control, but also stay undetected

“Evolutionary” insights (2)

- **Shorter generation lifetimes make for faster evolution**
  - Commodity systems and libraries shorten development time
  - Customization is costly, so we should expect less of it in the future
- **If software is embedded systems' “DNA”, then we should expect accumulation of passive, “unused” code**
  - Cutting away “unused” code is dangerous
  - More expensive to leave it in than to remove it
  - Developers need to “get the job done”, not optimize code footprint beyond what is necessary for the platform

“Invisible” devices

- **Not thought of as threats**
- **“Perceived innocence” abused**
  - DreamCast “DC phone home”, Higbee & Davis, BlackHat/Defcon 2002
  - Fake UPS, Spide~1, AutoNiN & Mystic, Defcon 2003
  - **Printer/copier/fax** combinations, **FX & Phenoelit**, BlackHat Europe 2003

+


What changed in recent years

- **An embedded system can be compromised and fully controlled by the adversary, yet continue to provide its intended functionality flawlessly.**
  - “Only looking at it when it breaks” is no longer a safe way

If [an embedded system] does have a **remote management interface**, the opponent of skill focuses on that and, once a break is achieved, will use those self-same management functions to ensure that **not only does he retain control over the long interval but, as well, you will be unlikely to know that he is there.**

Dan Geer @ SourceBoston, March 2009

Could this be a “bot”?

[redacted]

“A multimedia set-top box”

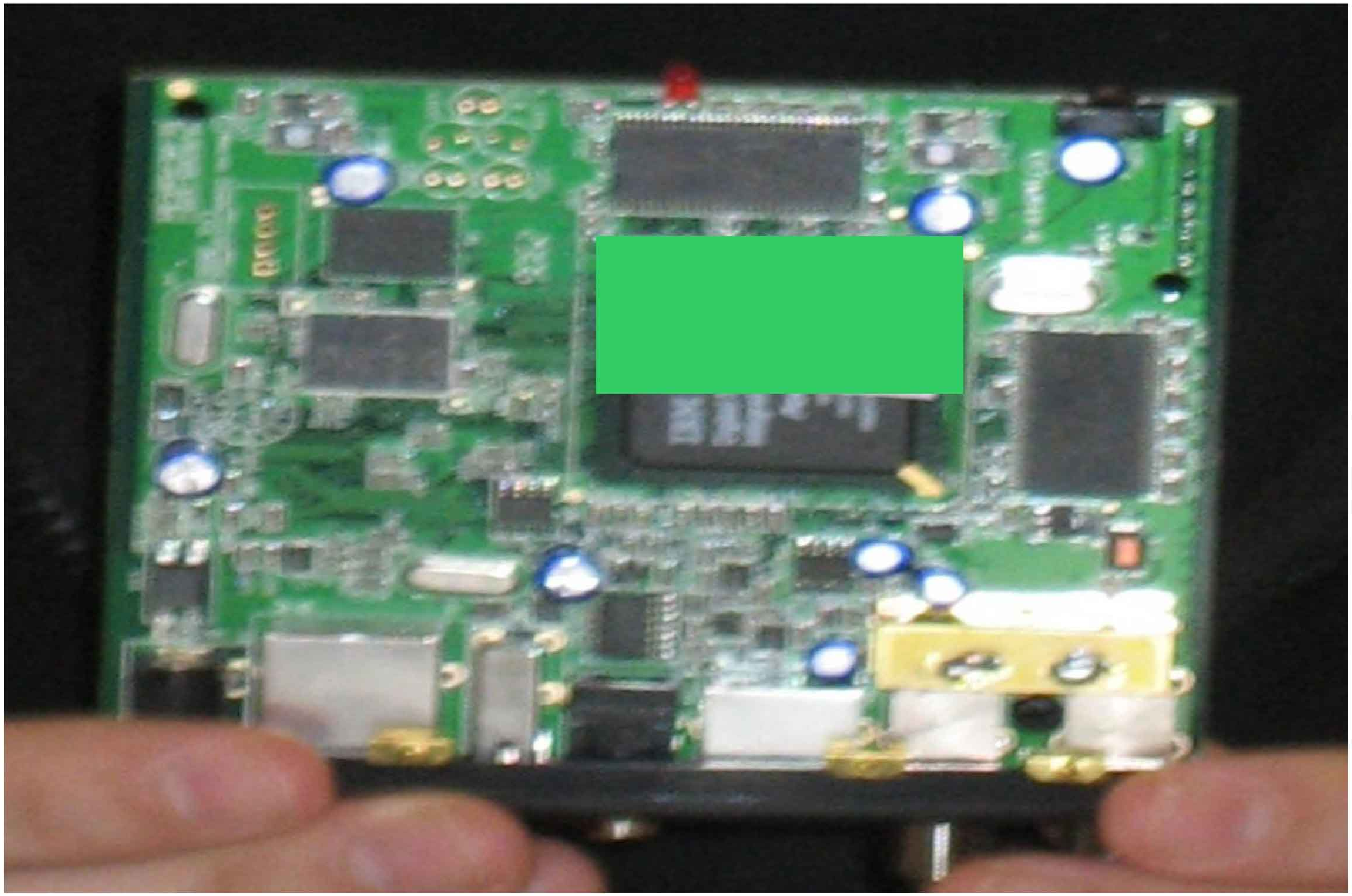
[redacted]

[redacted]

ERNW Living Security.

DARTMOUTH 1769

“Networked set-top box”


[redacted]

The story

- **Found during a network security audit**
  - It pays to Nmap your whole network ;-)
- **(Was) widely deployed on campus**
  - approximately 400 units
  - dorms, offices, lecture halls, ...
  - simple port scanning signature
  - intended to be zero-configuration (“just plug it into any wall jack”)
  - 100 Mbit Ethernet interface
- **Shell access for remote management**
  - Now it gets interesting :-)
  - Posted product manuals tend to contain default password...

Remove the cover ...

- **Linux 2.4 + BusyBox shell + custom applications**
  - Telnet server for admin access
  - Minimal web server for configuration
  - Most shell commands removed (e.g., no *chmod*)
  - Patched wget (used in scripts for downloading s/w updates)
- **Kernel configured to reduce memory footprint**
  - Boot loader unpacks root file system image from SDRAM, mounts it on a RAM disk
  - About 100K free "disk space"
- **Apparently hardened to limit root power**
  - Blocks most obvious ways to download and run an external executable

... void the warranty!

- **A stripped-down kernel?**
  - Still a commodity stock kernel, designed to fit many needs and uses
  - Squeezing the kernel down beyond removing unneeded devices is **hard**, even for Linux
    - **Who has the time to explore all options to remove?**
    - **Don't you want to just cut it down enough to fit your hardware, and get to work on programming the actual functionality?**
  - Easier to leave options in than remove them
    - **Who wants to do extra testing?**
- **Too much customization defeats the purpose of using a commodity kernel**
  - If you have to go through the kernel configs with a fine-toothed comb, where are the savings?

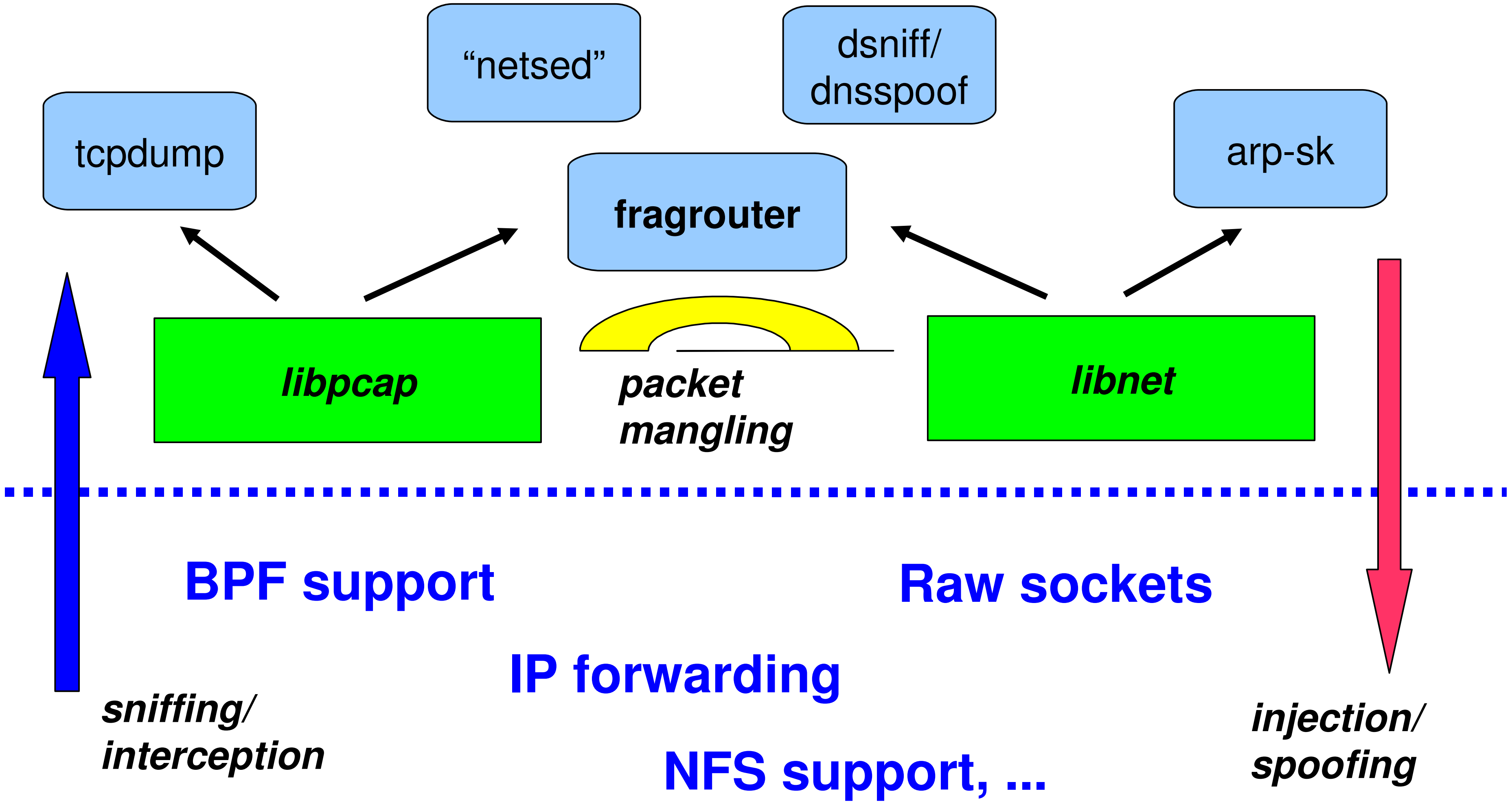
What we found in the kernel

- **Networked File System support**
  - Mount NFS shares over the network
  - Now we can attach any filesystem to read, write or execute!
- **Packet capture: Berkeley Packet Filter (BPF)**
  - Sniffing with libpcap (**tcpdump** & friends)
- **Raw IP and Ethernet socket support**
  - Inject crafted packets into the network
  - Now ARP poisoning is easy :-)
  - Add all the goodness of **Dsniff** + **Fragroute** (extended, more on this later)
- **IP forwarding (/proc/sys/net/ipv4/ip_forward)**
- **No IPtables/Netfilter** (so no QUEUE & inline packet editing) **:-(**

Some assembly required

- **Cross-compiling for the platform (IBM STBxxxx)**
  - GCC back end + ported libc
- **Libraries:**
  - Libpcap -- we want tcpdump, fragroute
  - Libnet -- we want to send spoofed ARP (**arp-sk**), DNS
  - Libdnet -- for **dsniff** goodness
    - SSL MITMs are old but old does not mean inefficient
  - OpenSSL – for SSL proxying to our box (with **socat**) and MITMs
  - Libevent – user-land packet forwarding and filtering (patched **fragroute**)
- **Tools**
  - Arp-sk, tcpdump, fragroute, netcat, socat, netsed, ...

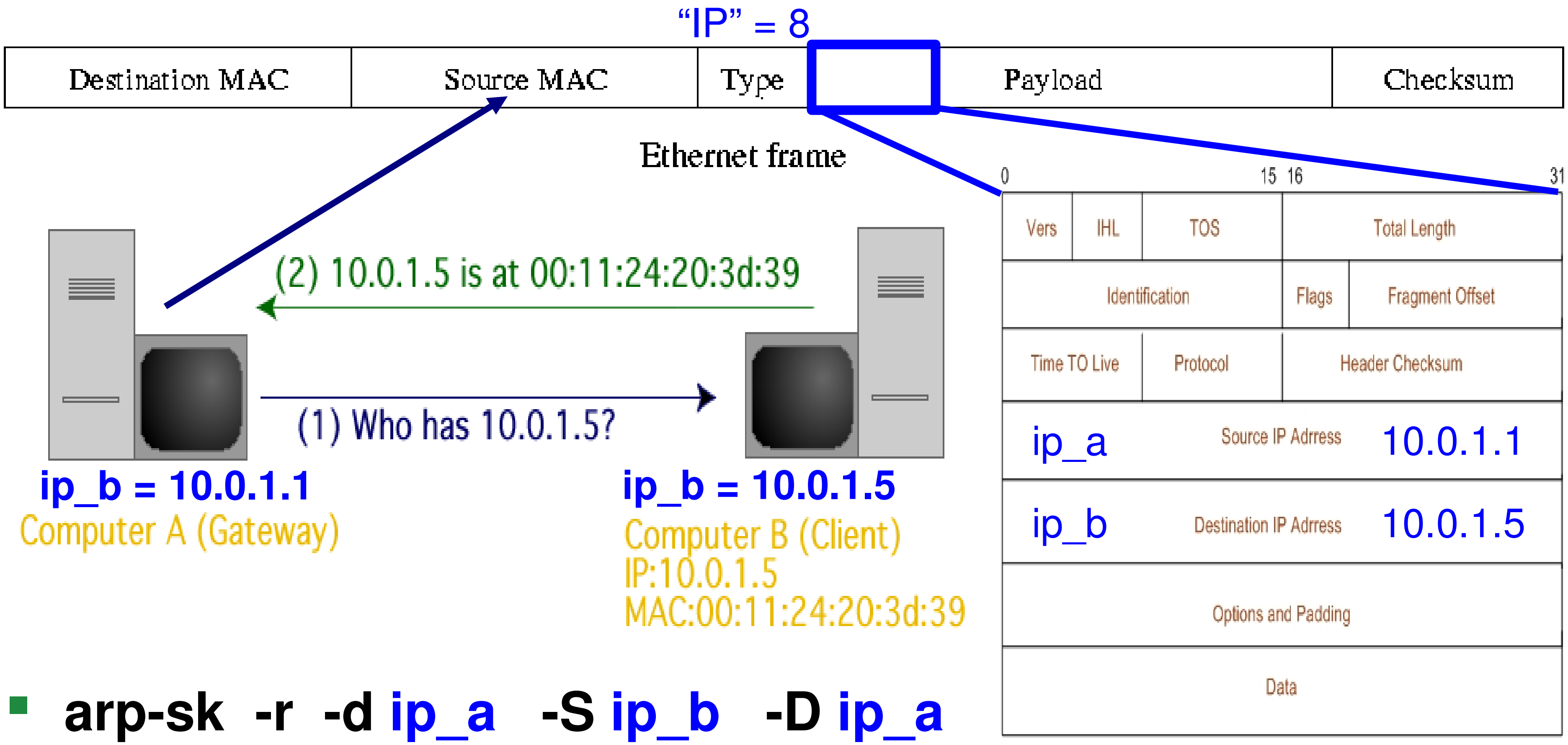
Attack tool chain at a glance



Oldies but goodies

- **“All your packets are belong to us”**
- **Sniff switched networks**
  - **Arp-sk** + mount NFS + **tcpdump** -s0 -w /mnt/remote-nfs/packets.pcap
- **DNS spoofing + connection forwarding**
  - Want to check your account balance?
- **Man-in-the-Middle TCP**
  - “Why did my images turn into goatses?”
- **IDS evasion via IP fragmentation**
  - That's what **fragroute** is for
- **Tunneling every which way**
  - **Socat** can forward almost anything (TCP, UDP, HTTPS, ...)

ARP poisoning for MITM

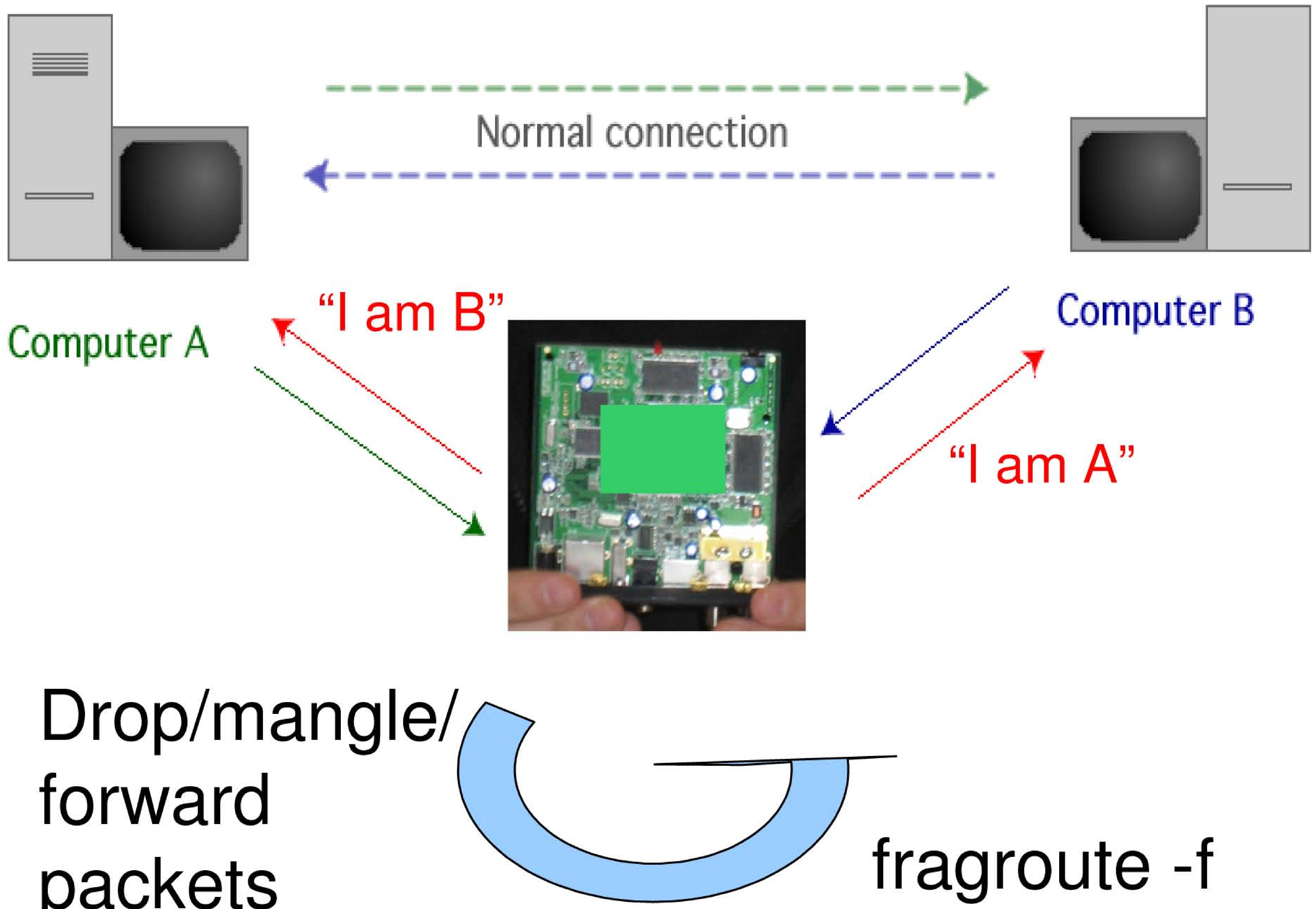


- arp-sk -r -d ip_a -S ip_b -D ip_a
- arp-sk -r -d ip_b -S ip_a -D ip_b

ERNW Living Security.

DARTMOUTH 1769

MITM with user-level routing



Drop/mangle/
forward
packets

fragroute -f
pcap_filter_exp

- **Need to drop some packets, rewrite some others, forward the rest**
- **IPtables would be nice, but not available**
- **Adapt <u>fragroute</u> instead – add filtering**

Hello old friend :-)

- **“[redacted] acquires [redacted] for 6.8 Billion Roubles”**
  - Subscriber equipment as shown on the vendor's website
  - This is *probably* the patched next version of this device

Online recent news headline

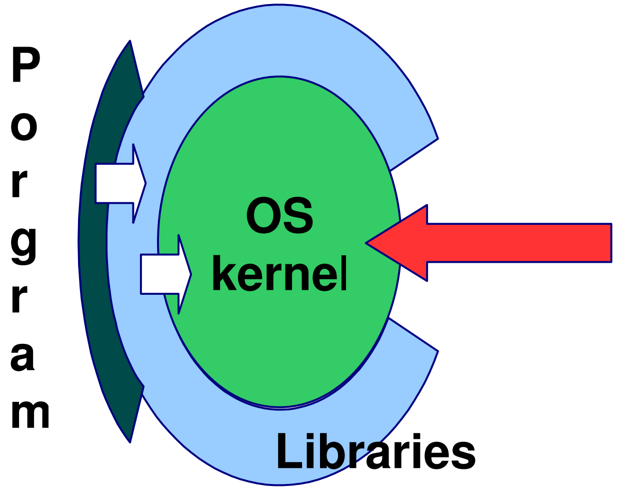
[redacted] [redacted]

How hard is this to fix?

- **Trim all extra functionality from the OS kernel?**
  - Requires broad and general knowledge of the kernel architecture
  - 2000+ **CONFIG_*** options in /usr/src/linux/.config for Linux 2.4
  - Can't mix-and-match Windows kernel functionality
- **Remote management interfaces must be present for longer-lived devices**
  - See Dan Geer's points on embedded systems as organisms
  - Crypto authentication for huge swarms of devices is hard (key management problems, “PKI”-type costs)
- **Devices must be self-protecting**

SELinux?

- **Removing library support does not prevent shellcode and exploits from making respective system calls**



```
mov  eax,  0x36  ; ioctl
int     0x80
...
mov  eax,  0x66  ; socketcall
int     0x80
```

- **Deny disallowed system calls based on process type:**
  - Track processes by ancestry and binary executable from **init**(1)
  - Only allow the minimal required set of system calls for each type

Dynamic tracing of programs?

- **Solaris's DTrace: system-wide tracing of selected programs and events with C-like scripts (the "D" language)**
  - Intercept ("probe") system calls and match their arguments
  - System and other process contexts available for checking at probe time
  - A script can keep state related to a process or thread
  - Executed in kernel, asynchronously
  - Very low performance penalty (designed for profiling production systems)
- **Linux is catching up with Kprobes + SystemTap**
- ***Trace-based "watchdog":*** **a system-wide script to check for conditions that "absolutely should not happen", kill processes or reboot the OS otherwise.**

Crazy Coke Machine

- **RISKS Digest 1996 (Peter Neumann)**
  - “Phone call deluge from program bug in computerized Coke machines”
  - “Another Coke machine phones for help, gets Fort Bragg number”

- **But these days they use TCP/IP ;-)**

Infected Digital Picture Frame

- A “digital panel” computer on the wall
- Used as student project showcase

- Source of mysterious, persistent port scans
- Hard to track
- No legitimate laptops or desktops in the area
- Guess what?

Infected Digital Picture Frame

- A “digital panel” computer on the wall
- Used as student project showcase

- Source of mysterious, persistent port scans
- Hard to track
- No legitimate laptops or desktops in the area
- Guess what?
  Windows CE, long unpatched

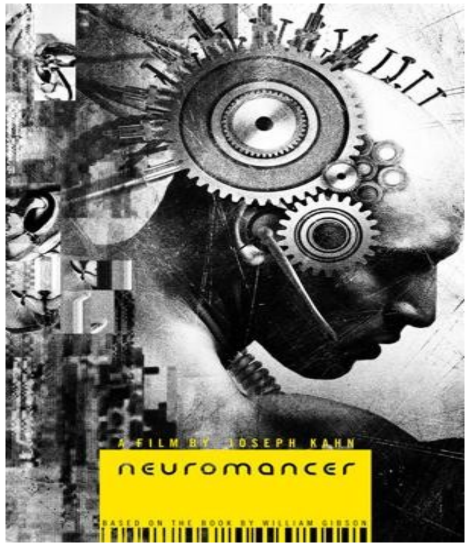
Next Generation Embedded Systems...

Next Generation Embedded Systems...

- **And now, think about it: what can those “devices“ do when abused...**

Next Generation Embedded Systems...

- **And now, think about it: what can those “devices“ do when abused...**



“Neuromancer”, 1984
William Gibson

Conclusions & Outlook

- **“Embedded” no longer means “too specialized too allow reliable, transparent hacks”**
  - “You don't have to be a desktop to be a zombie” (but it helps)
- **Watch out for the “invisible” devices**
  - Look for them when scanning your network
  - Know their OS and patch status
  - Find out their remote management capabilities before some else does
- **If you build Linux embedded systems:**
  - Strip the kernel of the juicy networking features, OR
  - Consider using **SELinux**: although the functionality is still there, it will be much harder to get to
  - Consider using a system-wide event tracer as a watchdog

Thank you!

ERNW
Living Security.

DARTMOUTH 1769