pin2pwn: How to Root an Embedded Linux Box with a Sewing Needle

Brad Dixon - Carve Systems
DEF CON 24
“USEFUL NOVELTY”

- It works
- Easy
- Teachable
- Dramatic

- Risky
- Crude
- Perhaps redundant
Demo

Actual backing tool
Prior Art

• Significant body of work around fault injection and glitching at the IC level for secure processors

• Recent system-level applications:
  - 2004: WRT54 “Bricked Router” recovery, Administrator note by mbm
  - “How to Hack the Hudl – We give Rockchip a good seeing to”, Pen Test Partners blog post
  - “20 Devices in 45 Minutes”, CJ Heres et. al., DEF CON 22 (related)
  - “WINKHUB Side Channel Attack”, Kevin2600, 2016
  - “Getting Root on a Philips Hue Bridge”, Colin O’Flynn, 2016
For today...

- **When** this attack can be effective
- **Why** this attack works
- **How** to defend against this attack
RISKS TO HARDWARE

• I have not yet destroyed hardware but this is abuse of semiconductor devices.

• Use on equipment you can afford to destroy.

• Depending on the hardware you may have better and safer options. Use those first.
Order of Attack
1. Serial
2. JTAG
3. …
4. Flash to CPU interface
Why does this work?

- Disrupt boot chain with a transient fault
- Activate an unexpected failure path
Scenario #1: Exploitable U-Boot Configuration

1. No JTAG.
2. Homegrown “secure” boot
3. Try to load and boot kernel #1
4. Try to load and boot kernel #2
5. If that fails then… return to U-Boot prompt!
Scenario #2: Exploitable Init Configuration

- `/bin/init` reads `/etc/inittab`
- `/bin/init` runs `/etc/rc`
- `/etc/rc` starts application in the foreground
- Application grabs console and presents a login prompt with credentials we don’t know
- BUT… if the application fails to load then `/bin/init` runs `/bin/sh`
How To Using LTE Router #4
How To

Prepare
• Survey HW
• Identify ports to monitor boot
• Datasheets
• Inspect failure modes, if possible
• Get boot timing

Poke
• Select pins to poke
• Get some timing visibility
• Poke!
• May take a few attempts
• Power-off between tests

Pwn?
• Monitor for unusual behavior
  • Serial traffic
  • Fallback boot configurations
  • Re-activated JTAG
  • Boot from TFTP
  • Fail to USB DFU
  • New network ports
• Sometimes you get lucky!
**pin2pwn rampage results**

Note: Table indicates pin2pwn vulnerabilities only

<table>
<thead>
<tr>
<th></th>
<th>Device</th>
<th>“secure” boot</th>
<th>Flash Type</th>
<th>uboot shell</th>
<th>root shell</th>
<th>Defense</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LTE Router #3</td>
<td>No</td>
<td>Serial</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LTE Router #4</td>
<td>No</td>
<td>Parallel</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>&lt;redacted&gt;</td>
<td>Yes</td>
<td>Parallel</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>&lt;redacted&gt;</td>
<td>No</td>
<td>Serial</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>LTE Router #5</td>
<td>No</td>
<td>Parallel</td>
<td></td>
<td></td>
<td>BGA</td>
</tr>
<tr>
<td>6</td>
<td>LTE Router #6</td>
<td>Yes</td>
<td>Parallel</td>
<td></td>
<td></td>
<td>Hash check</td>
</tr>
<tr>
<td>7</td>
<td>Home Automation Hub</td>
<td>No</td>
<td>Parallel</td>
<td></td>
<td></td>
<td>BGA, Fast</td>
</tr>
</tbody>
</table>
Defense: FAIL CLOSED

- Test your failure paths including transient hardware failure.
- Modify boot loaders to reboot at the end of the automated boot sequence.
- Enable watchdog time in bootloader, service in userspace.
- Be cautious shipping “fail to debug mode” features in production configurations.
Defense: Hide your pins and traces

- BGA surface mount devices hide their pins under the package
- Takes away the easy places to poke
- Make sure to route using inner layers

vulnerable traces under soldermask

poke where?
Defense: Run silent, run fast

- Very terse serial output.
- Fast kernel boot (0.1332 seconds) makes it sort of hard to jam the pin in there at the right time.
Thank you