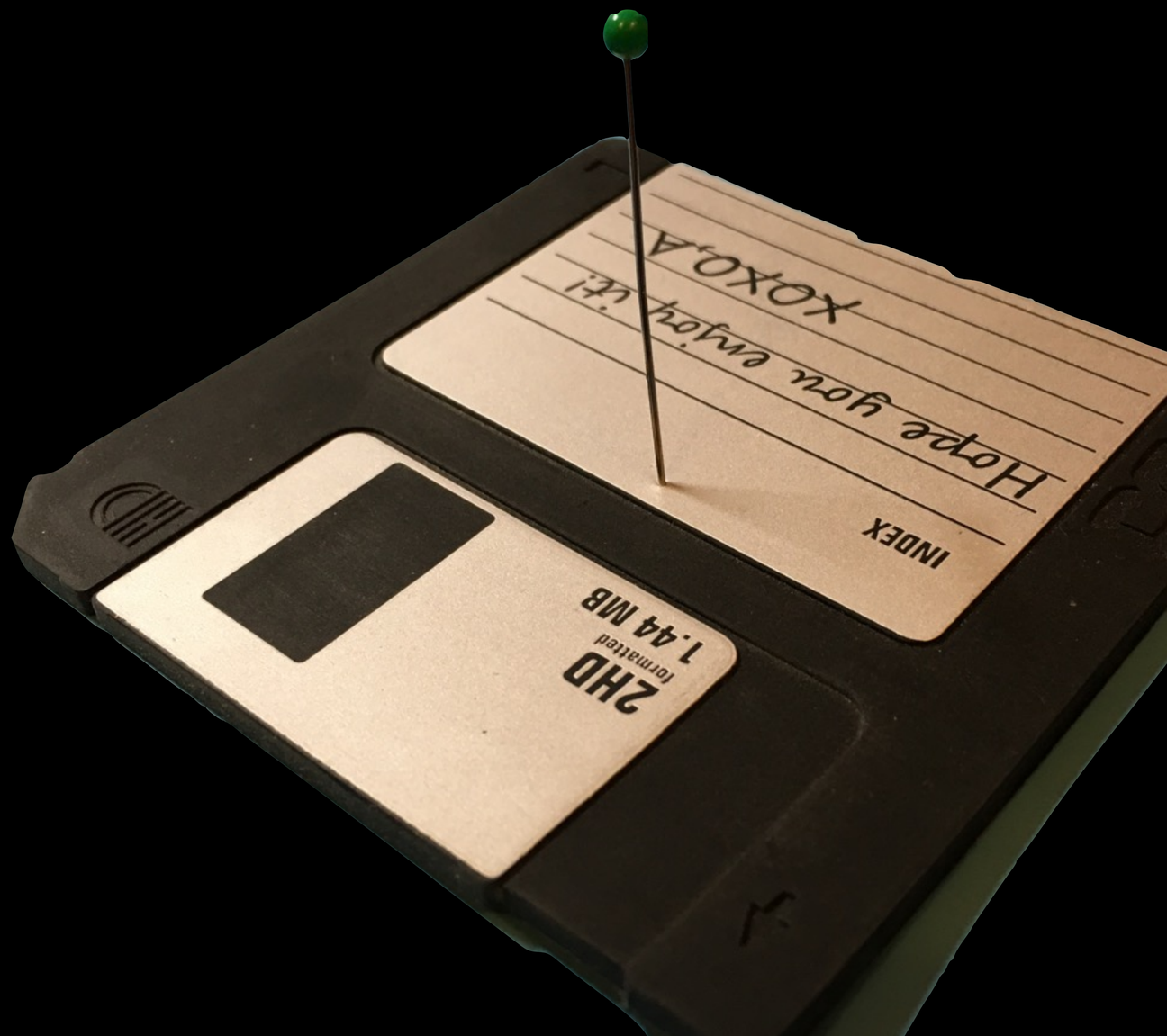
pin2pwn: How to Root an Embedded Linux Box with a Sewing Needle

Brad Dixon - Carve Systems

DEF CON 24



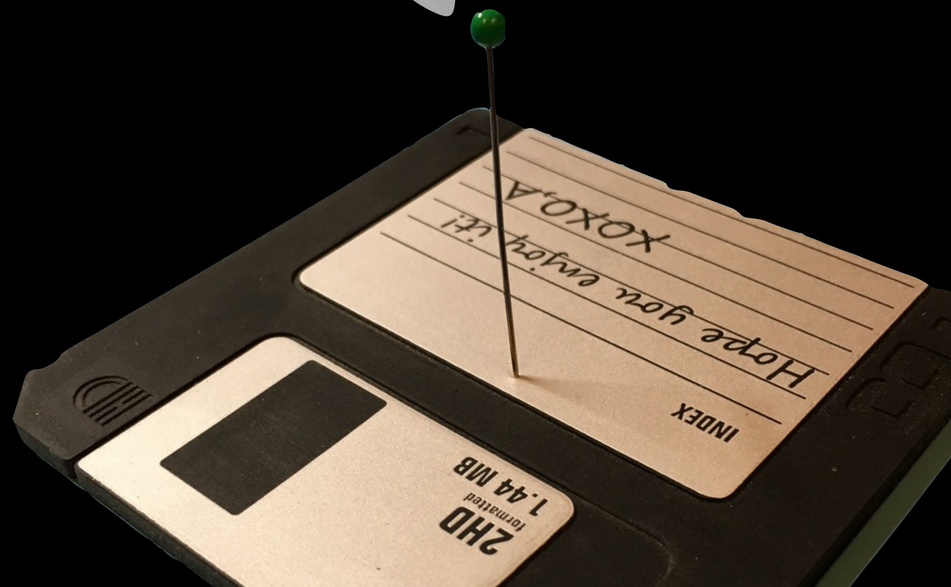
“USEFUL NOVELTY”

- It works
- Easy
- Teachable
- Dramatic

- Risky
- Crude
- Perhaps redundant

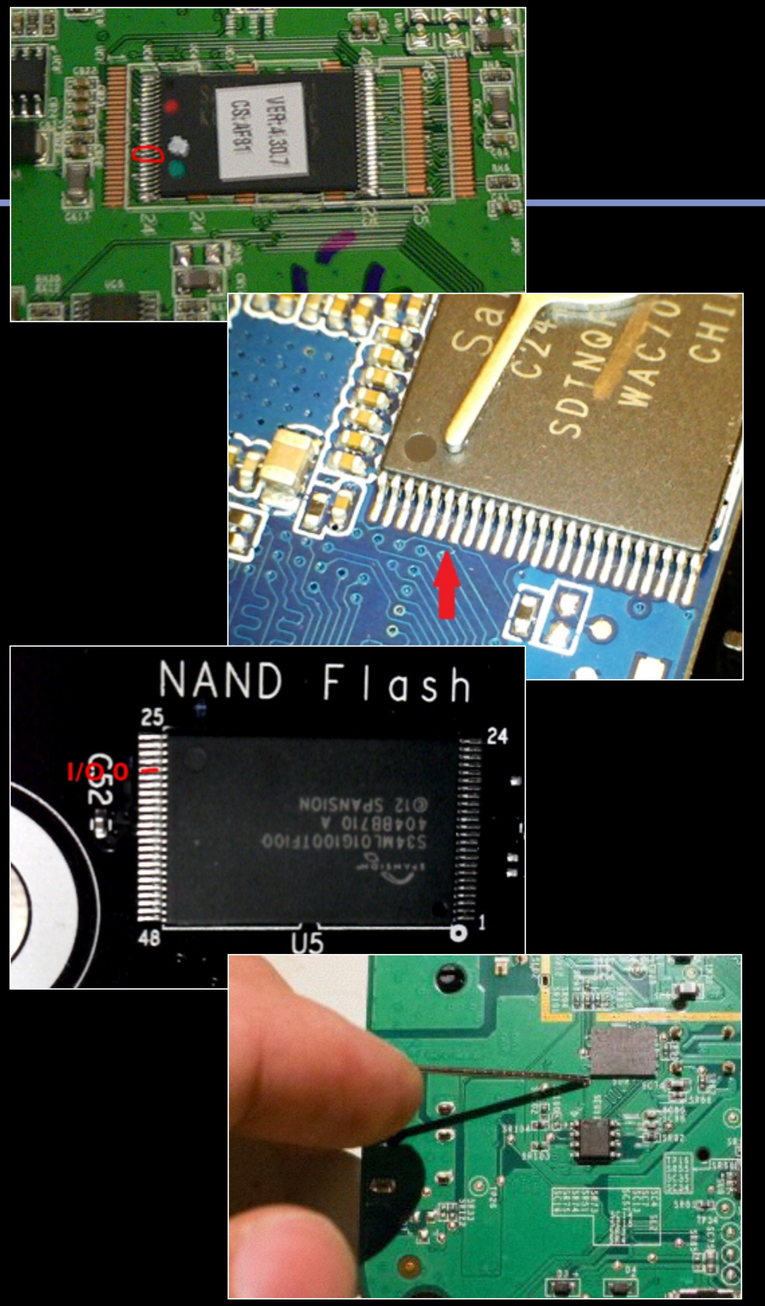
Demo

Acting / backing tool



Prior Art

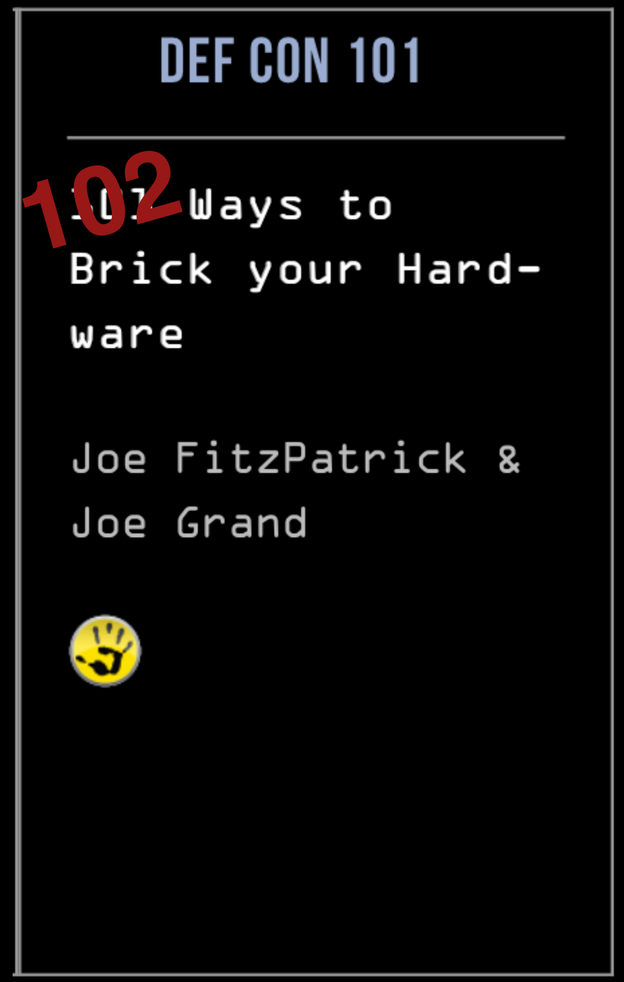
- Significant body of work around fault injection and glitching at the IC level for secure processors
- Recent system-level applications:
  - 2004: WRT54 “Bricked Router” recovery, Administrator note by mbm
  - “How to Hack the Hudl – We give Rockchip a good seeing to”, Pen Test Partners blog post
  - “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

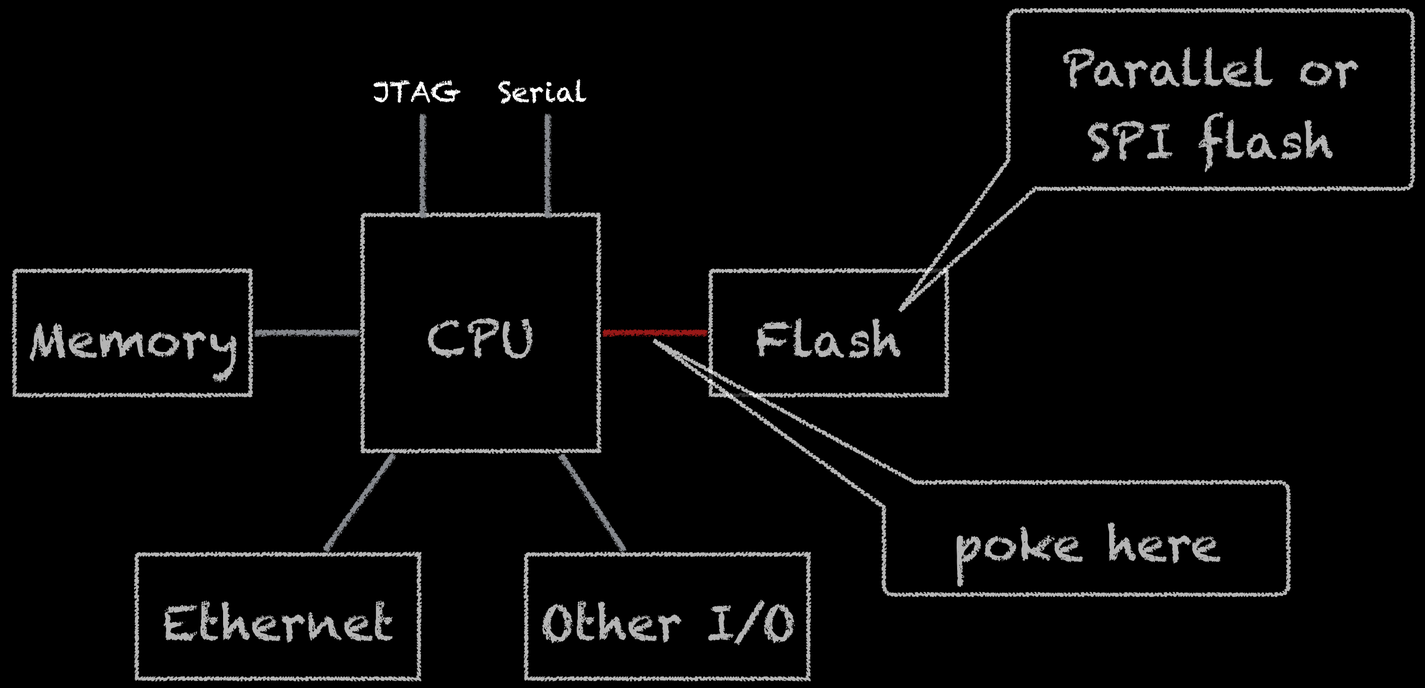


- I have not ***yet*** destroyed hardware but this is abuse of semiconductor devices.
- Use on equipment you can afford to destroy.
- Depending on the hardware you may have better and safer options. Use those first.

Generic Networked Doohickey Product Design

Order of Attack

1. Serial
2. JTAG
3. …
4. Flash to CPU interface

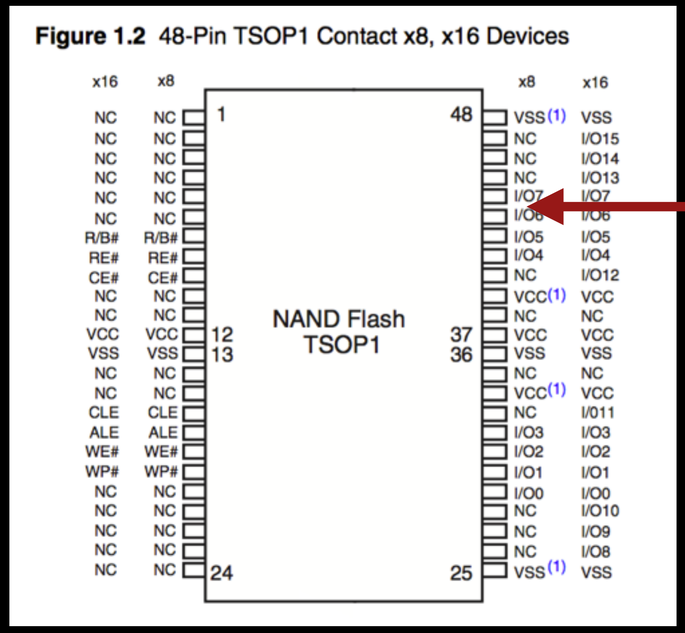


Why does this work?

- Disrupt boot chain with a transient fault
- Activate an unexpected failure path

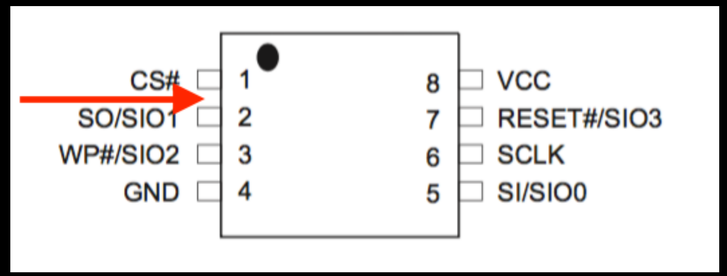
Scenario #1: Exploitable U-Boot Configuration

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!

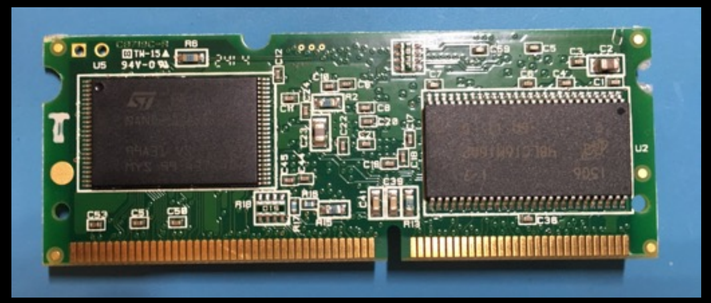


Scenario #2: Exploitable Init Configuration

- `/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



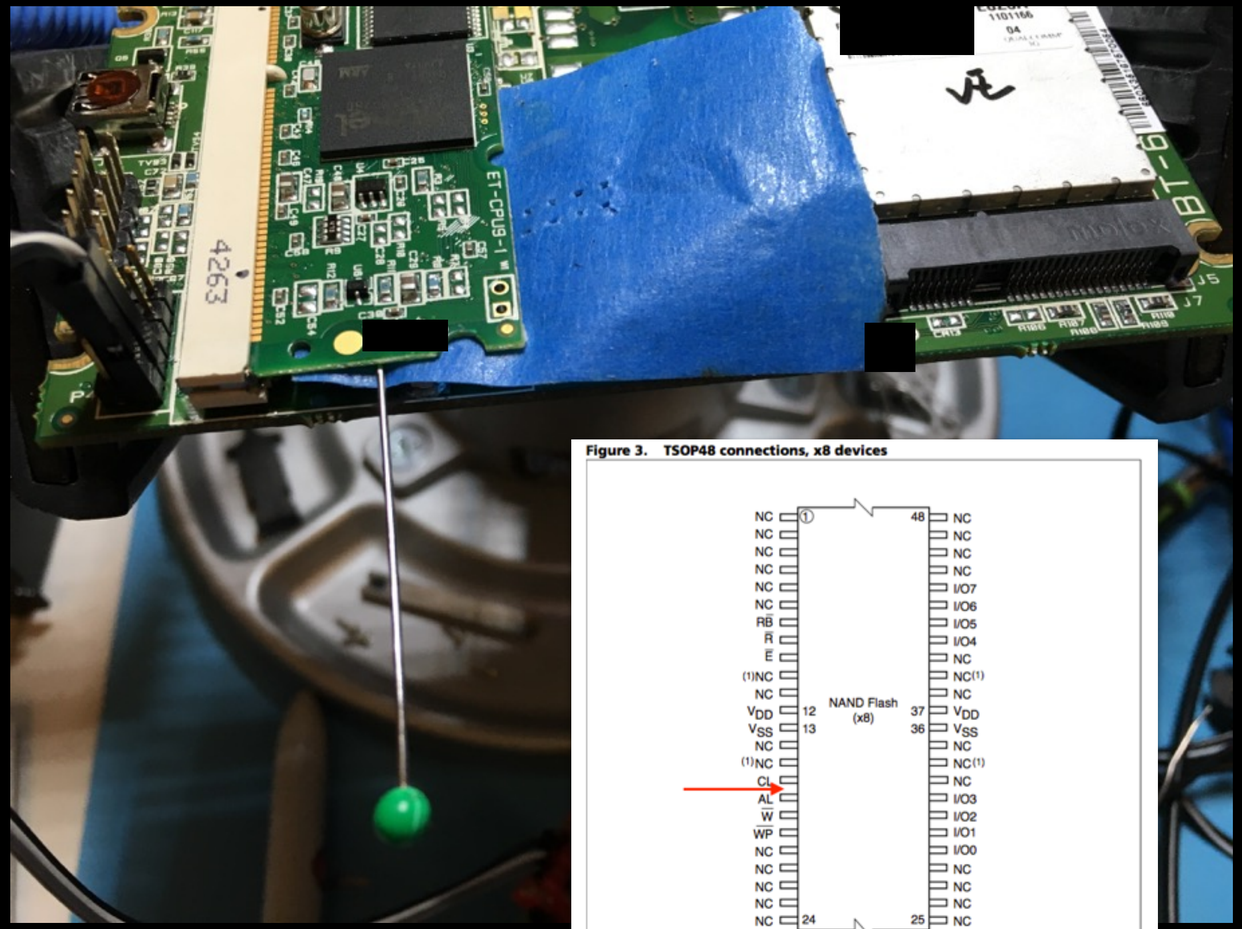


Figure 3. TSOP48 connections, x8 devices

1. This pin is DU in the USOP48 package

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!

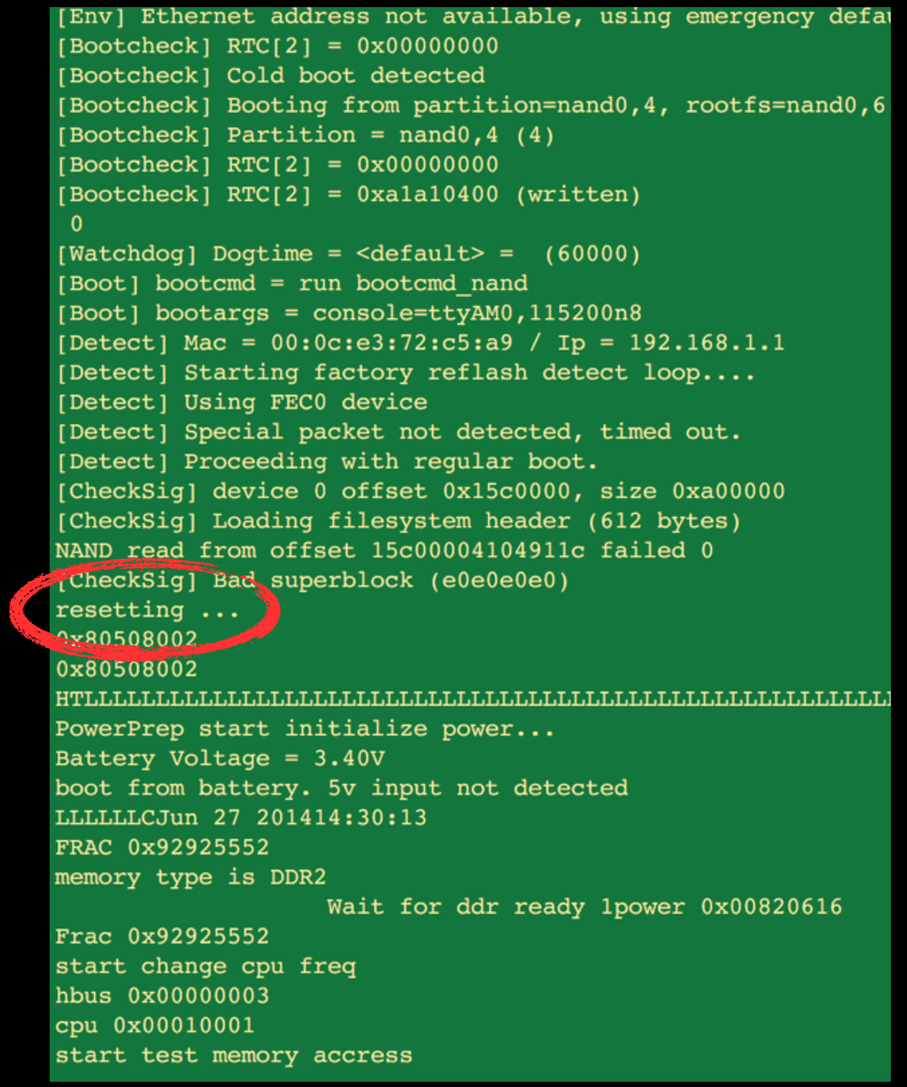
pin2pwn rampage results

Note: Table indicates pin2pwn vulnerabilities only

	Device	“secure” boot	Flash Type	uboot shell	root shell	Defense
1	LTE Router #3	No	Serial	✔		
2	LTE Router #4	No	Parallel	✔		
3	<redacted>	Yes	Parallel	✔		
4	<redacted>	No	Serial		✔	
5	LTE Router #5	No	Parallel			BGA
6	LTE Router #6	Yes	Parallel			Hash check
7	Home Automation Hub	No	Parallel			BGA, Fast

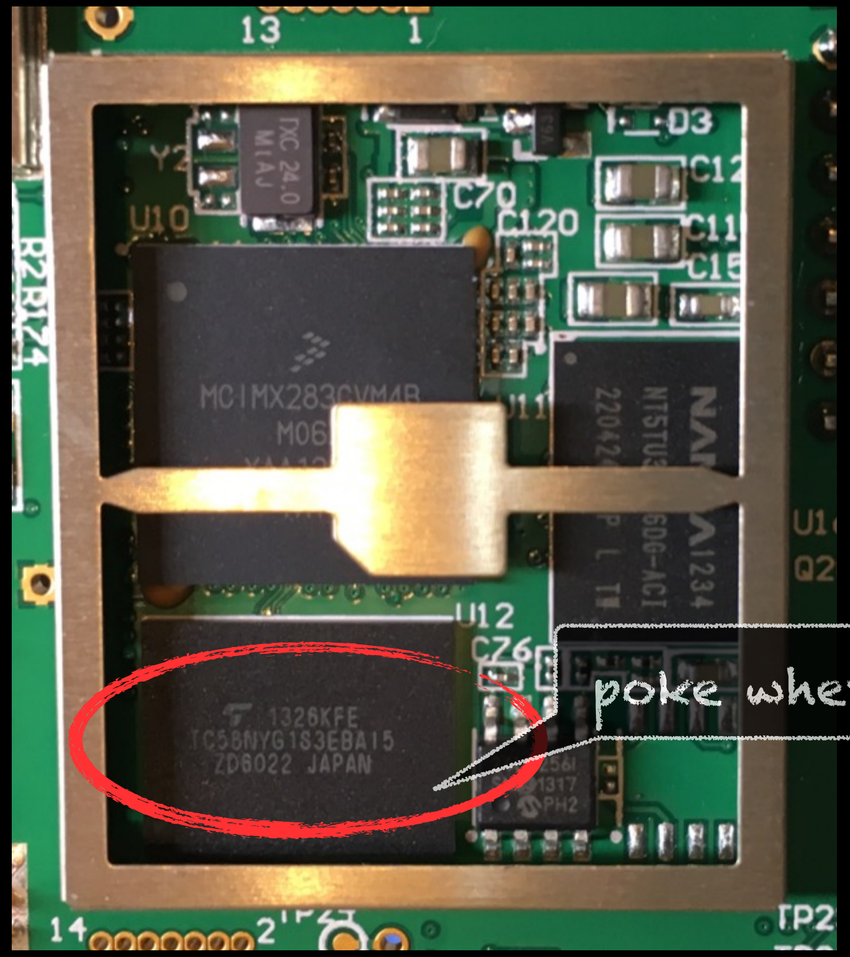
Defense: FAIL CLOSED

- 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.

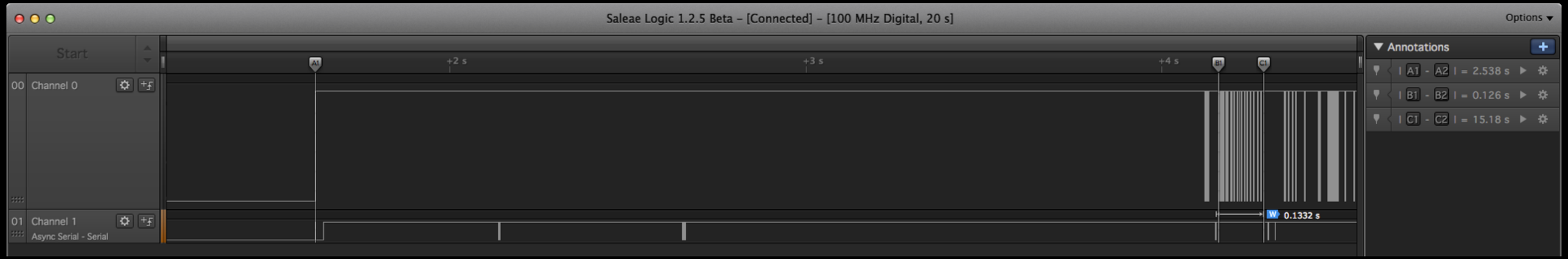


Defense: Hide your pins and traces

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



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

CARVE SYSTEMS