ECE 471 – Embedded Systems Lecture 19

Vince Weaver

http://www.eece.maine.edu/∼vweaver

vincent.weaver@maine.edu

11 November 2014

Announcements

- HW8 is out.
- Project topics

HW7 review

Embedded Systems security

- Until recently not an issue. Systems were embedded, so hard to compromise
- Now everything has a connection to the internet (cell phones, video games, thermostats, etc)
- Even systems that don't, can have publicly available access like USB ports and such

How to Find Bugs?

- Source code inspection
- Watching mailing lists
- Static Checkers
- Dynamic Checkers
- **Fuzzing**

Fuzzing

- 1988. Barton Miller. Noticed unix programs crash due to line noise. General case OK, but die if you feed them out-of-range data.
- Idea is to check for potential crashes in programs by feeding them random inputs and see how they handle it.
- Trinity for Linux (by Dave Jones) is one current project
- perf_fuzzer (by me) is another

- Often the best way to get crashes isn't truly random inputs, but almost-correct inputs

perf_fuzzer

- Wrote my fuzzer in response to a perf_event bug found using trinity (I had contributed perf_event support to trinity).
- That bug was a 64-bit config value was only checked for validity with a 32-bit cast (so only bottom 32-bits). A user could then use the full 64-bit value to point to memory and increment values. The clever hacker managed to increment the IDT instruction vector table, point the undefined instruction interrupt to root shell

code, and then executed an undefined instruction.

- I wanted to see if perf_event (a research interest of mine) had any other vulnerabilities.

Fuzzing – Bugs I found

- ARM config too big – the config field was used as an offset into a (small) array without checking the size. Kernel Panic

- ARM dangling pointer – a small struct with function pointers was cast to a larger struct with function pointers, and one of the function pointers off the end was called. Kernel Panic
  Also, by luck on very specific 3.11-rc kernels the dangling pointer pointed to 0x80000000, which is user mappable.

Put code there to set uid to 0 and exec a shell and you have root. (Got a CVE, but getting this fixed took forever. Luckily it's very unlikely anyone was ever affected by this).

- Have found other, more boring, bugs. Usually just computer lockups.

Case Studies

Examples – CANbus

- 2010 IEEE Symposium on Security and Privacy. *Experimental Security Analysis of a Modern Automobile* U of Washington and UCSD.
- Fuzzing/ARM/CANbus
- can control brakes (on / off suddenly)
- heating, cooling, lights, instrument panel
- windows/locks Why? fewer wires if on a bus then

direct-wired

- electronic stability control, antilock, need info from each wheel
- roll stability control (affect braking, turning to avoid rollover)
- cruise control
- pre-crash detection (tighten seatbelts, charge brakes)
- while it might be nice to have separate busses for

important and unimportant, in practice they are bridged

- Locks– monitor buttons, also remote keyfob... but also disengage if airbag deploys
- OnStar – remotely monitor car, even remotely stop it (in case of theft) over wireless modem
- Access? OBD-II port, also wireless
- 2009 car
- cars after 2008 required to have canbus?

- Problems with CAN
  - Broadcast... any device can send packets to any other
  - Priority.. devices set own priority, can monopolize bus
  - No authentication... any device can control any other
  - Challenge-response. Cars are supposed to block attempts to re-flash or enter debug mode without auth. But, mostly 16-bits, and required to allow a try every 10s, so can brute force in a week.
  - If you can re-flash firmware you can control even w/o ongoing access

- Not supposed to disable CAN or reflash firmware while car moving, but on the cars tested they could.
- Probing – packet sniffing, fuzzing (easier as packet sizes small)
- experiments – on jackstands or closed course
- controlled radio – display, sounds, chimes
- Instrument panel – set arbitrary speed, rpm, fuel, odometer, etc

- Body control – could lock/unlock (jam by holding down lock), pop trunk, blow horn, wipers on, lights off
- Engine... mess with timing. forge ”airbag deployed” to stop engine
- Brakes.. managed to lock brakes so bad even reboot and battery removal not fix, had to fuzz to find antidote
- can over-ride started switch. wired-or
- test on airport. cord to yank laptop out of ODB-II

- fancy attacks. Have speedometer read too high. Disable lights. ”self-destruct” w countdown on dash, horn beeping as got closer, then engine disable.

Stuxnet

- SCADA – supervisory control and data acquisition
- industrial control system
- STUXNET.. targets windows machines, but only activates if Siemens SCADA software installed. four zero-day vulnerabilities
  USB flash drives
  signed with stolen certificates

- Interesting as this was a professional job. Possibly by US/Israel targeting very specific range of centrifuges reportedly used by Iran nuclear program. While reporting "everything OK" the software then spun fast then slow enough to ruin equipment.

Examples – JTag/hard-disk

- JTAG/Hard-disk takeover
- `http://spritesmods.com/?art=hddhack&page=8`
- Find JTAG
- 3 cores on hard-disk board, all ARM. One unused.
- Install custom Linux on third core. Then have it do things like intercept reads and change data that is read.

Other worrisome embedded systems

- Backdoors in routers.
- Voting Machines, ATMs
- pacemakers
- Rooting phones
- Rooting video games