ECE 471 – Embedded Systems
Lecture 25

Vince Weaver

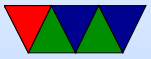
http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

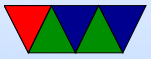
2 November 2022

Announcements

- Project topics are due, will respond to them

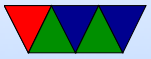


Coding Mistakes with Security Implications



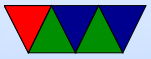
Dangling Pointer / Null Pointer Dereference

- Typically a NULL pointer access generates a segfault
- If an un-initialized function pointer points there, and gets called, it will crash. But until recently Linux allowed users to `mmap()` code there, allowing exploits.
- Other dangling pointers (pointers to invalid addresses) can also cause problems. Both writes and executions can cause problems if the address pointed to can be mapped.

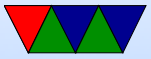


Privilege Escalation

- If you can get kernel or super-user (root) code to jump to your code, then you can raise privileges and have a "root exploit"
- If a kernel has a buffer-overrun or other type of error and branches to code you control, all bets are off. You can have what is called "shell code" generate a root shell.
- Some binaries are setuid. They run with root privilege but drop them. If you can make them run your code before dropping privilege you can also have a root exploit.



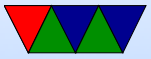
- ping (requires root to open raw socket)
- X11 (needs root to access graphics cards)
- web-server (needs root to open port 80).



Information Leakage

- Can leak info through side-channels
- Detect encryption key by how long other processes take? Power supply fluctuations? RF noise?
- Timing attacks
- If code takes different paths through code can notice this via linked info

  Solution: cycle-invariant code, takes same amount of time for all paths through code (really hard to write code like this)

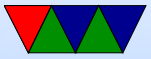


Information Leakage: Meltdown and Spectre

- Can use timing to find if address is in cache
- If speculative execution, can do things like

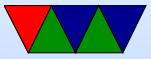
```
if (secret&1) a[0]=1;
else a[4096]=1;
```

then use timing to see which one was brought in



Deceptive Code

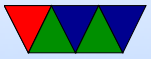
- Can you sneak purposefully buggy/exploitable code into open source?
- Can you sneak bad code (or use typo-squatting) to trick people in large public repositories (like javascript/npm)
- To-do at U of Minnesota where researches tried (unsuccessfully it turns out) to sneak questionable code into the kernel
- "Trojan Source" in the news: can use unicode (including



left-right reversal) to have code that looks correct but compiler will compile differently

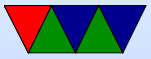
- Should code allow non-ASCII?

  to apply updates



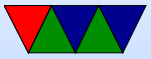
Finding Bugs

- Source code inspection
- Watching mailing lists
- Static checkers (coverity, sparse)
- Dynamic checkers (Valgrind). Can be slow.
- Fuzzing



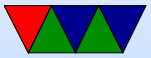
perf_fuzzer

- Fuzzers intentionally try invalid/dangerous input by generating random inputs causing crash
- I wrote the `perf_fuzzer` which found many bugs in Linux kernel with the perf_event_open() syscall

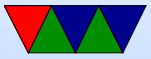


Reporting Bugs

- So you found a security bug...
- Who do you contact?
- What's responsible disclosure?
- Bug bounties
- Can be a hassle reporting properly, and companies are always suspicious and can even accuse you of evil hacking

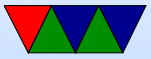


Computer Security



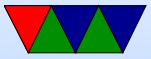
Social Engineering

- Often easier than actual hacking
- Talking your way into a system
- Looking like you know what you are doing
- “The Art of Deception”

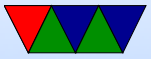


Worrisome embedded systems

- Backdoors in routers.
- Voting Machines, ATMs
- pacemakers – what stops someone from updating firmware?
- Rooting phones
- Rooting video games

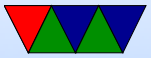


- Others?



Voting Machines

- Maine has paper ballot — not too bad
- Often are old and not tested well (Windows XP, only used once a year)
- How do researchers get them to test? e-bay?
- USB ports and such exposed, private physical access
- Can you trust the software? What if notices it is Election Day and only then flips 1/10th the vote from Party A to Party B. Would anyone notice? What if you have source code?



- What if the OS does it. What if Windows had code that on Election Day looked for a radio button for Party A and silently changed it to Party B when pressed?
- OK you have and audit the source code. What about the compiler? (Reflections on Trusting Trust). What about the compiler that compiled the compiler?
- And of course the hardware, but that's slightly harder to implement but a lot harder to audit.