ECE 471 – Embedded Systems
Lecture 24

Vince Weaver

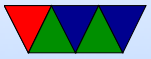
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

31 October 2022

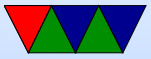
Announcements

- Don't forget HW#7 (SPI)
- Be sure to set frequency in the ioctl()
- Don't forget Project topics
  will respond to them via e-mail
- Why cast to unsigned long from pointer? Isn't that bad?
  Has to do with low-level Linux interface and 32/64-bit ioctl emulation



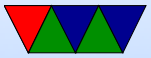
Computer Security
and why it matters for embedded systems

- Most effective security is being unconnected from the world and locked away in a box. Until recently most embedded systems matched that.
- Modern embedded systems are increasingly connected to networks, etc. Embedded code is not necessarily prepared for this.
- Internet of Things: IoT (the S is for Security)

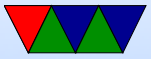


Big Event Where This Matters

- Election next Tuesday
- Places with Electronic Voting Booths
- Have been found trivial to hack. Running windows, with exposed USB connector.
- How did researchers get access to them. (eBay)
- Attacks often have to be local unless you happen to hack main database

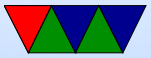


- Paper ballots tend to be more secure
- Social Engineering issues.
- What about vote-by-mail? Ruins anonymous voting, as people can bribe/watch you vote
- Internet voting (?!)



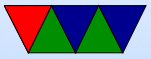
The Problem

- Untrusted inputs from user can be hostile.
- Users with physical access can bypass most software security.



What can an attacker gain?

- Fun / Mischief
- Profit
- A network of servers that can be used for illicit purposes (SPAM, Warez, DDOS)
- Spying on others (companies, governments, etc)

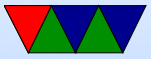


Sources of Attack

- Untrusted user input
  Web page forms
  Keyboard Input

- USB Keys (CD-ROMs)
  Autorun/Autostart on Windows
  Scatter usb keys around parking lot, helpful people plug into machine.

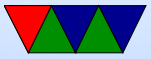
- Network



cellphone modems
ethernet/internet
wireless/bluetooth

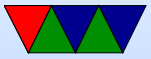
- Backdoors
  Debugging or Malicious, left in place

- Brute Force – trying all possible usernames/passwords



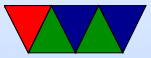
Types of Security Compromise

- Crash
  "ping of death"
- DoS (Denial of Service)
- User account compromise
- Root account compromise
- Privilege Escalation
- Rootkit
- Re-write firmware? VM? Above OS?



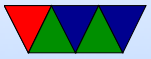
Unsanitized Inputs

- Using values from users directly can be a problem if passed directly to another process
- If data (say from a web-form) directly passed to a UNIX shell script, then by including characters like ; can issue arbitrary commands: `system("rm %s\n",userdata);`
- SQL injection attacks; escape characters can turn a command into two, letting user execute arbitrary SQL commands; xkcd `Robert '); DROP TABLE Students;--`



Buffer Overflows

- User (accidentally or on purpose) copies too much data into a fixed sized buffer.
- Data outside expected area gets over-written. This can cause a crash (best case) or if user carefully constructs code, can lead to user taking over program.

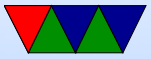


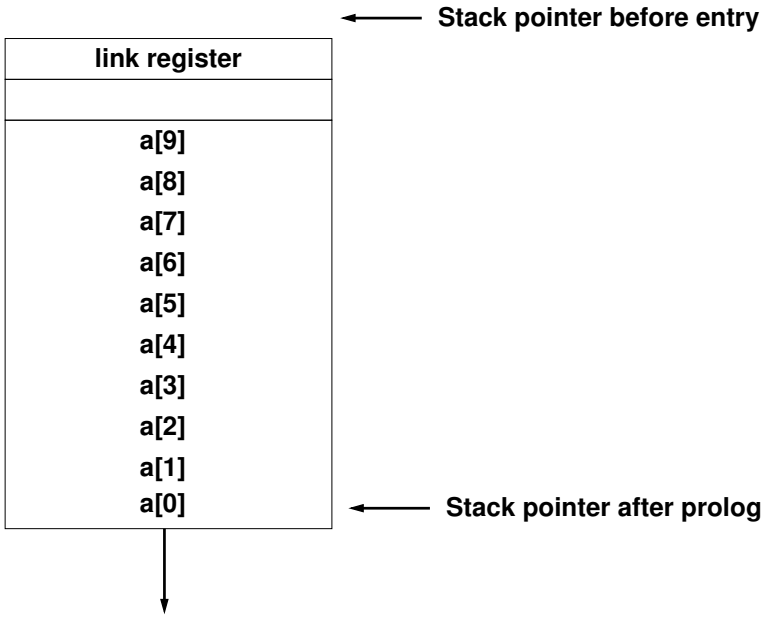
Buffer Overflow Example

```
void function(int *values, int size) {
    int a[10];

    memcpy(a,values,size);

    return;
}
```

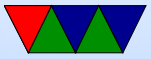
Maps to

```
    push    {lr}
    sub sp,#44

    memcpy

    add sp,#44
    pop {pc}
```





A value written to a[11] overwrites the saved link register. If you can put a pointer to a function of your choice there you can hijack the code execution, as it will be jumped to at function exit.



Mitigating Buffer Overflows

- Extra Bounds Checking / High-level Language (not C)
- Address Space Layout Randomization
- Putting lots of 0s in code (if strcpy is causing the problem)
- Scanning for unusual characters (can you write all-ASCII shellcode?)
- Running in a "sandbox"