Linux Privilege Escalation

Why?

- Usually get in with few perms
  - Webservers drop permissions
- You want control over the box
  - Pivoting
  - Takeover
  - Access to other local systems
- To look cool
  - “I got root” makes it more likely to be fixed (and more money)

Disclaimer

- PLEASE DON’T DO THIS WITHOUT FULL PERMISSION FROM THE OWNER
  - Used to be bigger because of mainframes
  - Computer misuse act basically was designed for this

Simple misconfiguration

- Very common
- Give users too much permission by default
  - sudo access
  - ssh access
  - db access

Less simple misconfiguration

- Trying to be secure, but failing
  - sudo, but only for a preset program
    - sudo -l
  - setuid binaries that aren’t secure
    - find / -perm /6000 -executable 2>/dev/null
  - Insecure default configuration / workarounds

Insecure sudo

- sudo can be restricted to only run certain programs
- Most programs aren’t designed to be secure as root
- sudo drops the LD_PRELOAD and PATH environment variables
- People often miss this:
  - https://security.stackexchange.com/q/233135
  - Demo
- Use gtfobins: https://gtfobins.github.io/

Insecure suid

- suid/sgid is a low-level permission
  - Allows the program to run as a different user
- Carries across environment variables (except for LD_PRELOAD)
  - Mess with PATH for easy root
- Exploits are mostly similar to sudo

Mess with $PATH for easy root

- How does a shell know what commands do?
  - When running a command, shells look for programs in $PATH
- Add your own directory, your ones will run instead
  - `mkdir /tmp/<whatever>`
  - `chmod 555 /tmp/<whatever>`
  - `export PATH="/tmp/<whatever>:$PATH"`
  - Now create a file with the same name, put "#!/bin/sh" as the first line, write some commands
  - Use chmod a+x <file> to mark it as executable
- This will work on anything that doesn't clean path, and then runs shell commands

Race conditions

- Program expects some state not to change, often the contents of a file
- Jump in between the actions
  - Usually between set and get
- Often pops up in surprising places
  - Dirty CoW vulnerability – pwn linux kernel

Misconfigured services

- Very context-dependent
- You’ll get this with experience
- But Google often is a useful substitute for experience :)

Idiocy

- Often people just leave passwords/backups lying around
- Often admins write tools with hardcoded creds to make their lives easier
- Don’t waste time when you can just easily grep for flags!