Computer Network Defense

we begin with the triad (aka, the things you should definitely do)
- patch updates
  - #1 most overlooked security technique
  - as vulnerabilities are found, patches are released
  - how long (on average) do you think a vulnerability exists before it is discovered?
    - 342-ish days (uhhh, yeah, almost one year!)
  - patches may make you safe from *most* attacks
    - just not 0-day attacks

- malware protection
  - no, you are not invincible!
    - no matter what you think

  - malware?
    - viruses
    - worms
    - bacteria
    - trojans
    - rootkits
    - spyware (e.g., sniffers, keyloggers, etc)
    - adware (e.g., annoying popups, spam, phishing, etc)

  - defense?
    - anti-virus
    - anti-spam/anti-adware
    - anti-malware?
      - hash detection, basically

- firewall
  - take care of what's on your system
  - two philosophies
    - block based on port (which usually ties to services/protocols)
      - any application attempting connection on a port is blocked
      - this is the Linux way
    - block based on application
      - a single application is blocked
      - this is the Windows way
      - usually means having to interact more with the firewall
        - which is usually annoying

defensive operations
- what can we do to “protect” ourselves?
- one option is to encapsulate our services/OS
  - e.g., virtualization (hypervisor and virtual machines)
    - e.g., virtualbox, vmware, xen, proxmox, etc
  - e.g., chroot jails (more on this later)
  - e.g., docker containers

- defense in depth
  - don't depend on a single mechanism for protection

layered approach (multiple layers of defense) – like an onion in your network
idea: use several varying methods

defense in breadth
there are many attack vectors (i.e., just having a firewall won't guarantee security)
we must try to cover all attack vectors

PDR^3: a framework for security
prevent → detect → respond → recover → restore → avoid (?) →

avoid
threat avoidance
threats simply don't matter
we don't care about detection, mitigation, prevention, attribution
we have an invisibility cloak
e.g., beaconing malware, unauthorized network users/apps, port knocking

offensive operations
reconnaissance and footprinting
useful to see if we might want to gain access to a system we don't have access to
recon: what's there? what systems exist?
footprinting: what specific things can we gather about those systems?
we might want to know a few things about a system:
what OS it runs
what hardware it has
what servers are running and on what ports
including versions of all of these (some may have known vulnerabilities!)

recon tools
nmap: security scanner for network exploration and security audits
nemesis: packet crafter and sender
python-scapy: packet swiss army knife
netcat: tcp swiss army knife
telnet: not as good as netcat

recon/footprinting tactics
port scanning
probes remote host for open ports
used to verify security policies and identify running services
portscan: scan for listening ports
portsweep: scan multiple hosts for a specific port
some worm may portsweep many hosts for a single port (vulnerability)
port status
open/accepted: something is listening
closed/denied/not listening: connection is denied
filtered/dropped/blocked: no reply (firewall?)

network sniffing (particularly under the same subnet) – “sniffer”
packet analysis
intercepts/logs network traffic (packets)
we can then decode/analyze these packets
uses:

analyze network problems
detect network intrusion attempts
gain info for possible network intrusion
monitor network usage
gather/report network stats
filter content from traffic
spy on users/collect sensitive information
reverse engineer proprietary protocols
debug client/server communication
debug network protocols