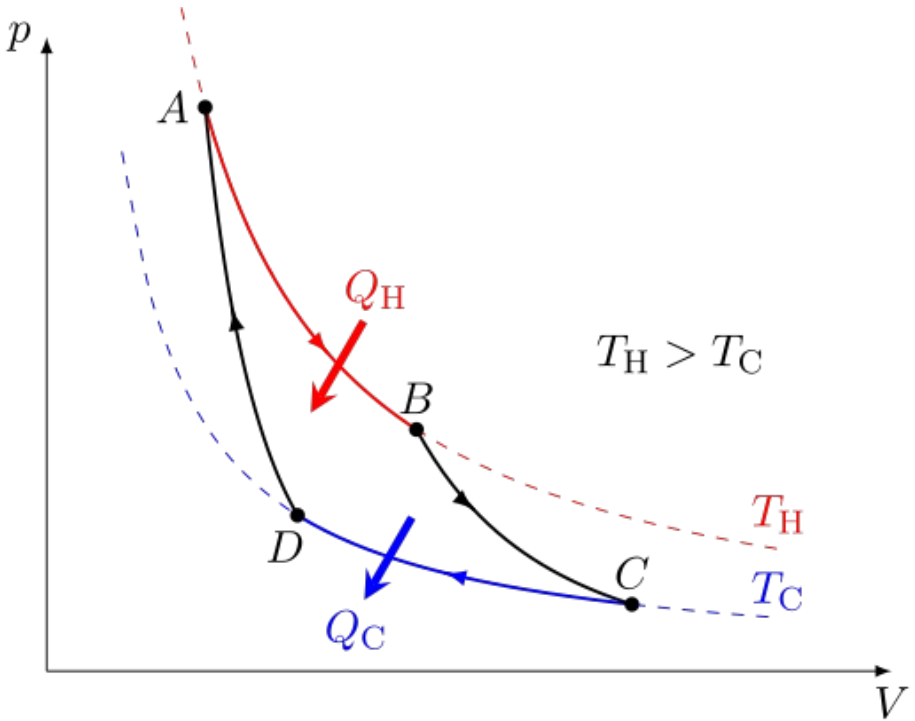
Port scanning and network side channels

CSE 548 Spring 2025

jedimaestro@asu.edu

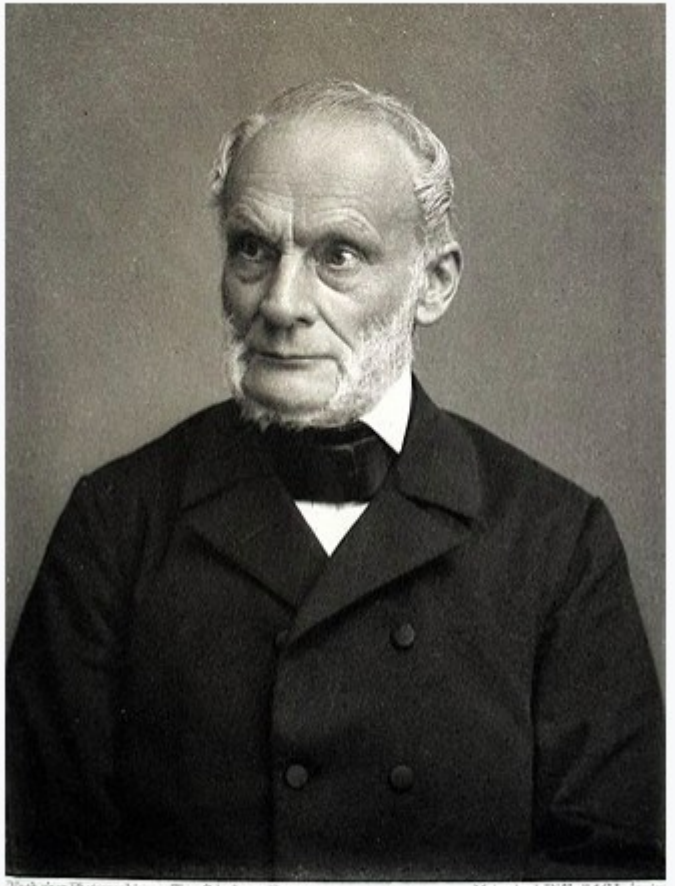
Main takeaways

- Expand your understanding of what a “bit” is
  - Information theory
  - Applies to NIDS evasion of encrypted traffic
- Understand why 2nd law of thermodynamics → shared, limited resources → side channels
- Understand that because shared, limited resources are everywhere in computer networks, so are side channels
  - And some side channel attacks can be very powerful



https://en.wikipedia.org/wiki/Nicolas_L%C3%A9onard_Sadi_Carnot
https://en.wikipedia.org/wiki/Carnot_heat_engine

Rudolf Clausius



“entropy”

(from Greek ἐν en "in" and τροπή tropē "transformation")

Like energy, but you can’t use it.

https://en.wikipedia.org/wiki/Rudolf_Clausius

Entropy

- Statistical foundation by Gibbs, Boltzmann, Maxwell, Planck, *etc.*
- Directly inspired the name of entropy in Shannon’s information theory

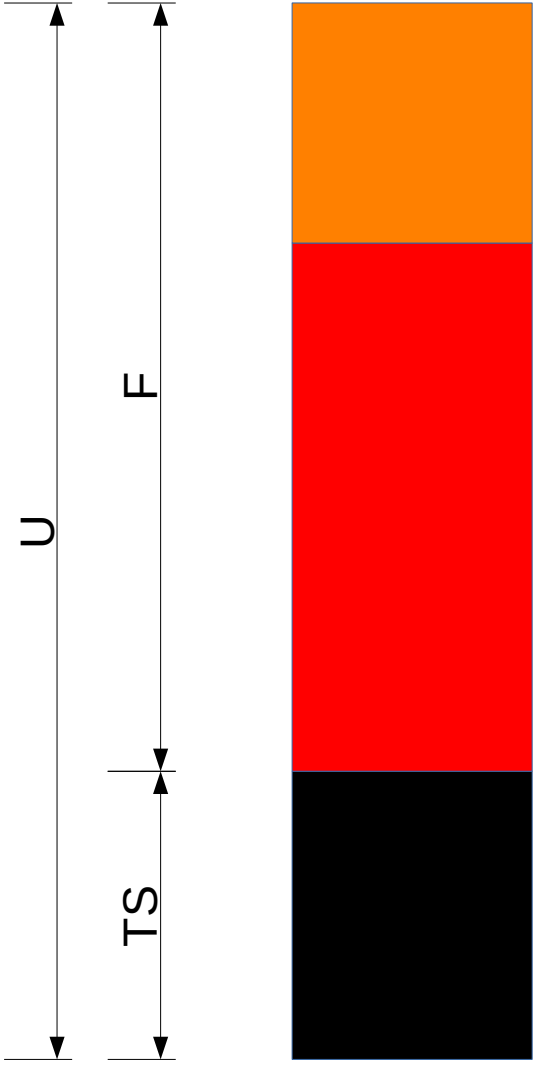
$$H = -\sum_i p_i \log_2 (p_i)$$

Reality

- Real engines aren’t as efficient as a Carnot engine
  - Efficiency of 20% or less, compared to 37% Carnot efficiency limit
    - https://news.mit.edu/2010/explained-carnot-0519
- Real computing devices and algorithms don’t use the available energy with 100% efficiency, either
    - Where does that energy go?

$$F = U - TS$$

F = free energy
U = total energy
T = temperature
S = entropy



energy necessary for the computation

energy “wasted”

lost to entropy

Resources are not free

- Must be shared
  - Buffers
  - Wires
  - Ports
  - Caches for fragments and segments
  - Rate limitations
  - …

How do we measure “information”?

- Entropy
  - Don’t be confused if you’ve heard this term in a physics class
    - Entropy in physics is the information we don’t have about energy, which leads to wasted energy
    - Entropy in information theory is a measure of how surprised we’ll be when we learn information, which leads to useful information

Requirements (Shannon, 1948)

1) $I(p) \geq 0$ (information is non-negative, $p \geq 1$)
2) $I(1) = 0$ (events that always occur carry no information)
3) $I(p_1 p_2) = I(p_1) + I(p_2)$ (information due to independent events is additive)

Also, continuity, symmetry, and maximum when all possible events are equiprobable.

$$I(p) = \log(1/p)$$

1) $I(1/2) = 0.30102999566...$
2) $I(1) = 0$
3) $I(1/2) + I(1/3) = \log(2) + \log(3) = 0.77815125038...$

Joint probability: $I(1/6) = 0.77815125038...$

Continuity: $I(1/2.01) = 0.30319605742...$

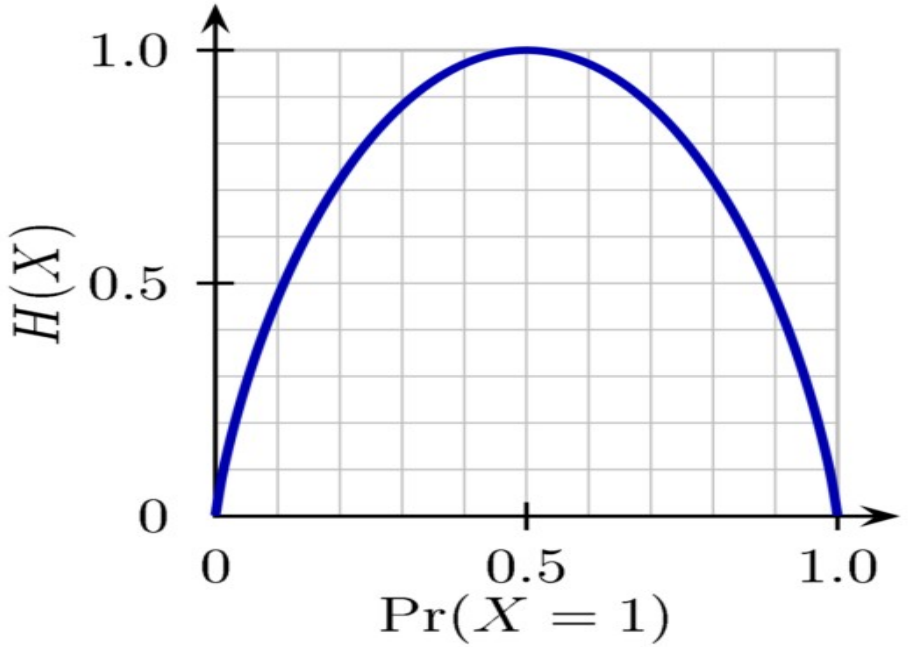
Symmetry: $\log(3) + \log(2) = 0.77815125038...$

Maximum: $\log 3 + \log 3 + \log 3 = 1.43136376416...$

$\log 2 + \log 4 + \log 4 = 1.20411998266...$

Information = Entropy = Surprise

$$H[p] = -\sum_{i=1}^{k} p_i \log p_i$$



Side note – Differential Entropy

https://en.wikipedia.org/wiki/Differential_entropy

$$h(X) = \mathbf{E}[-\log(f(X))] = -\int_{\mathcal{X}} f(x)\log f(x)\,dx$$

f(x) is a probability density function for the signal, the more the signal “jumps around” the higher the entropy, therefore modulating higher frequencies
means more entropy and therefore more bandwidth.

Not a pop quiz #1

- When a 3yo walks by with a stepstool…
  - 4 times out of 10 it’s to get something they’re not supposed to have
  - 2 times out of 10 it’s to climb up to somewhere they’re not supposed to be
  - 1 time out of 10 it’s to wash their hands
  - 1 time out of 10 it’s to get something they’re allowed to have
  - 1 time out of 10 it’s to use as a dollhouse
  - 1 time out of 10 it’s to turn over and use as a storage bin
- What is the entropy of each instance of 3yo stepstool habits?

Answer

Input:

$$-0.4 \log_2(0.4) - 0.2 \log_2(0.2) - 4\,(0.1 \log_2(0.1))$$

Result:

$$2.32193\ldots$$

Not a pop quiz #2

- There are three possible states the Tempe weather could be in during any given hour on a summer day (very hot and bright out, very hot and it’s nighttime, monsoonal rains). What probability distribution over these events would give the maximum entropy in terms of what you might observe in a randomly chosen hour from the summer?

Answer

Input:

$$-\frac{1}{3}\log_2\left(\frac{1}{3}\right)-\frac{1}{3}\log_2\left(\frac{1}{3}\right)-\frac{1}{3}\log_2\left(\frac{1}{3}\right)$$

Exact result:

$$\frac{\log(3)}{\log(2)}$$

$\log(x)$ is the natural logarithm

Decimal approximation:

More digits

1.584962500721156181453738943947816508759814407692481060455...

Not a pop quiz #3

You have 12 coins, one is counterfeit. The counterfeit is either slightly heavier or slightly lighter, otherwise it’s impossible to tell. You have a balance. Using the balance the fewest number of times, find the counterfeit coin.

Definitions

- *Covert channel*: a channel two processes can use for communication that was not intended to be used for communication
  - Sender and receiver collude
- *Side channel*: a channel through which information leaks, but the sender is not sending the information intentionally
  - No collusion

Outline

- Review of port scanning, idle scans
- Examples of ***network*** side channels
  - SYN backlogs and DoS
  - RST rate limitation
  - Counting packets off-path
  - Off-path TCP hijacking

TCP 3-way handshake (review)

- SYN: I'd like to open a connection with you, here's my initial sequence number (ISN)
- SYN/ACK: Okay, I acknowledge your ISN and here's mine
- I ACK your ISN

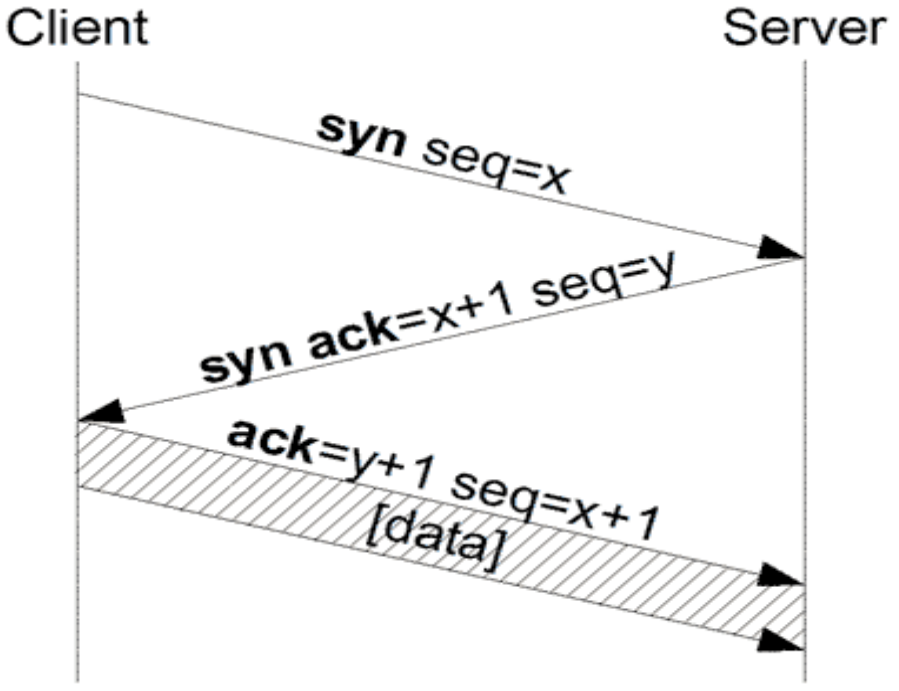


Image from Wikipedia

Open port == listening

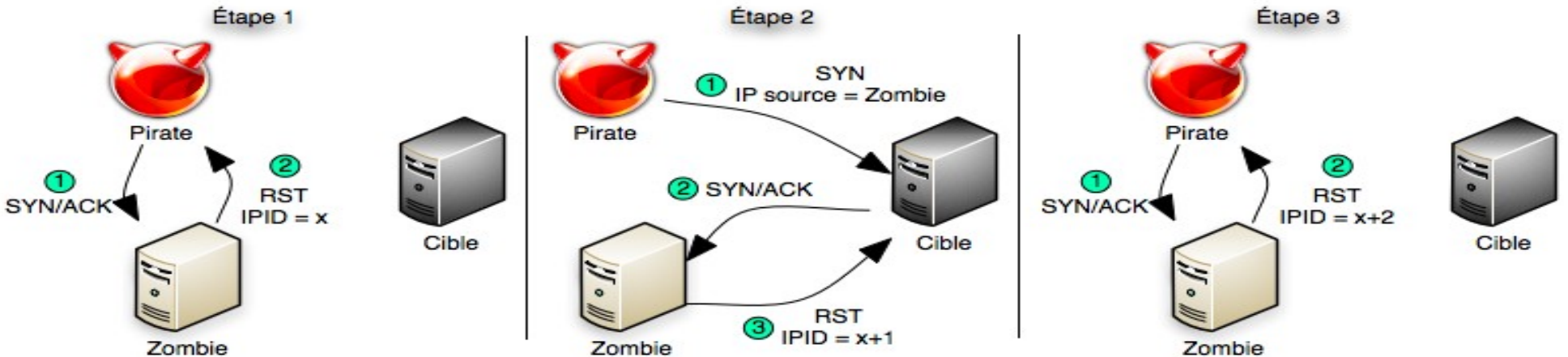
- If you send a SYN packet to port 80 (the HTTP port) on a remote host and that host replies with a SYN/ACK, then we say that port 80 on that machine is “open”
  - In this example, that probably means it's a web server
- If it responds with a RST, we say it's “closed”
- If there is evidence of filtering (no response or ICMP==Internet Control Message Protocol error), we say it's “filtered”
  - UDP is more complicated: open|filtered *vs.* closed

Things nmap can do

- Is a port open? Closed? Filtered?
  - Many ports on one machine is a “vertical scan”
- For a /24 network, which machines are up? Which machines have port 80 open?
  - One port for a range of machines is a “horizontal scan”
- OS detection (research on your own)
- Stealth, info about middleboxes, etc.

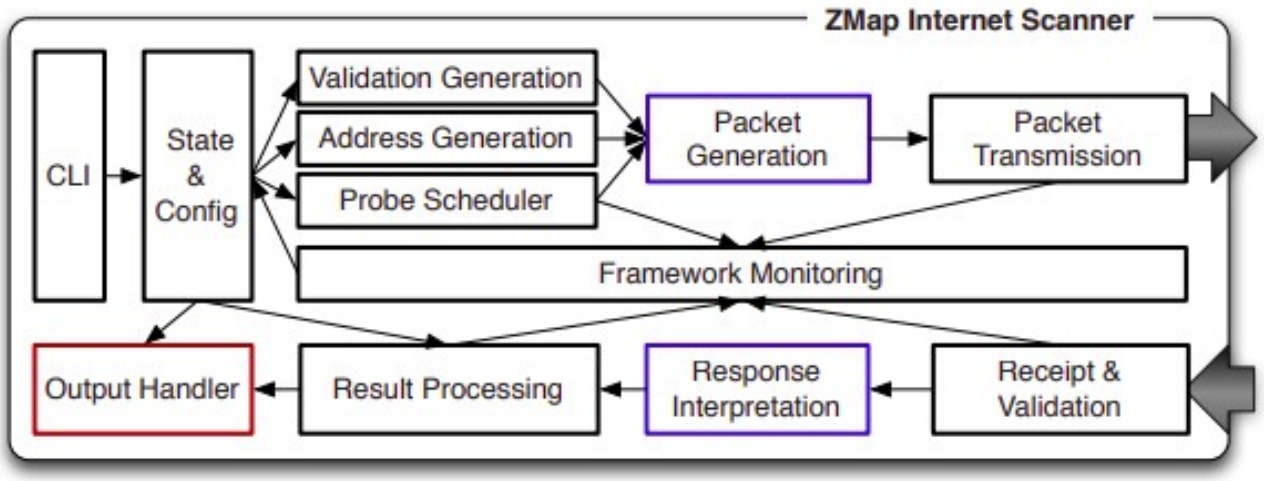
Idle scan

- Every IP packet sent has an IP identifier
  - In case it gets fragmented along the way
- Old machines (or just that are configured that way) use a globally incrementing IPID that is shared state for all destinations



Zmap

- https://www.usenix.org/system/files/conference/usenixsecurity13/sec13-paper_durumeric.pdf
- https://zmap.io/



Theme

- Attacker wants to find out (*i.e.*, copy) certain information while doing the least amount of work possible
  - Are hackers lazy, or do they just respect the 2nd law of thermodynamics?
    - Yes, both
  - Copying information is the simplest computation you can do, and is what reversible molecular computers use as a benchmark
  - Side channels (like the idle scan) are the same thing, just more indirect...

More examples of network side channels...

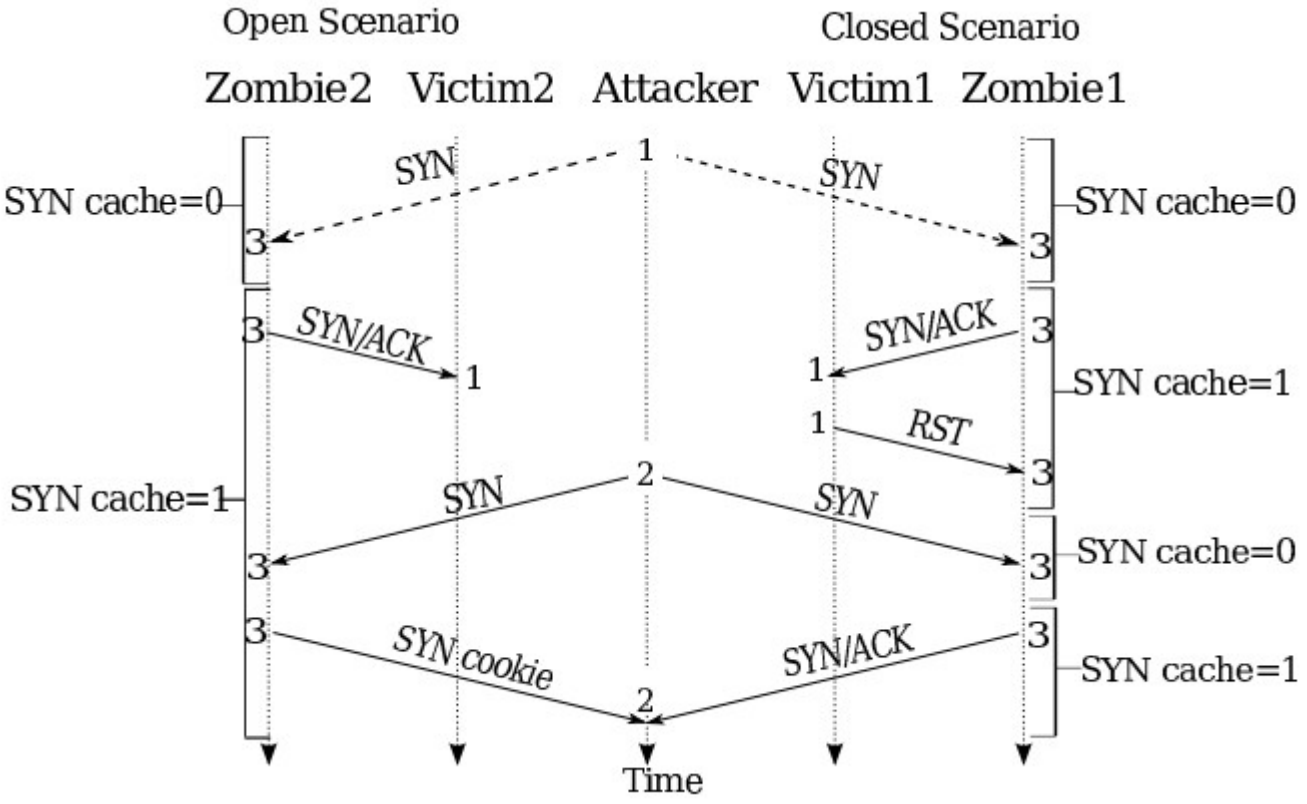
- DoS and SYN backlog basics
  - A side channel based on the SYN backlog
- Counting packets off-path
- Blind off-path TCP hijacking

DoS in general

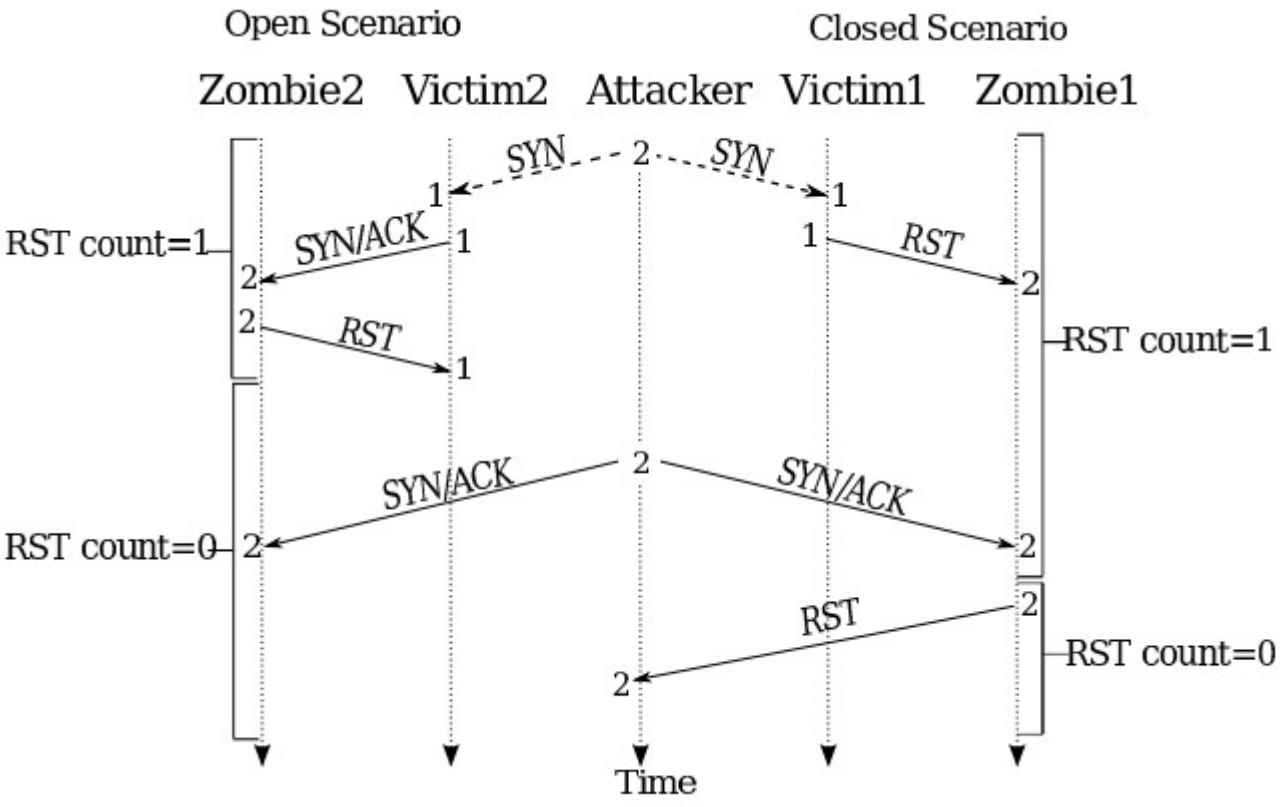
- Exhaust some kind of resource, *e.g.*:
  - Optimistic ACK to exhaust bandwidth
    - See https://homes.cs.washington.edu/~tom/pubs/CCR99.pdf
  - PING of death (*e.g.*, large PING) causes crash
  - Exhaust CPU in layer 7
  - More examples: http://www.isi.edu/~mirkovic/bench/attacks.html
  - SYN flood: Older hosts had either a fixed amount of half-open connections they could keep track of or no limitations at all; attack is to send lots of SYNs and never ACK or RST
    - Defenses: SYN backlog policies and SYN cookies

SYN cookies and SYN backlogs

- SYN cookies
  - Special kind of SYN/ACK
  - See https://cr.yp.to/syncookies.html
  - Can confirm ACK number and reconstruct the necessary state for a connection without having kept any state after sending the SYN cookie
- SYN backlog examples
  - Linux reserves ½, ¼, 1/8th, and so on for successively older SYNs, prunes 5 times a second
  - FreeBSD has 512 buckets of 30, you can't predict what bucket you fall into (in theory)



From… https://jedcrandall.github.io/usenix10.pdf



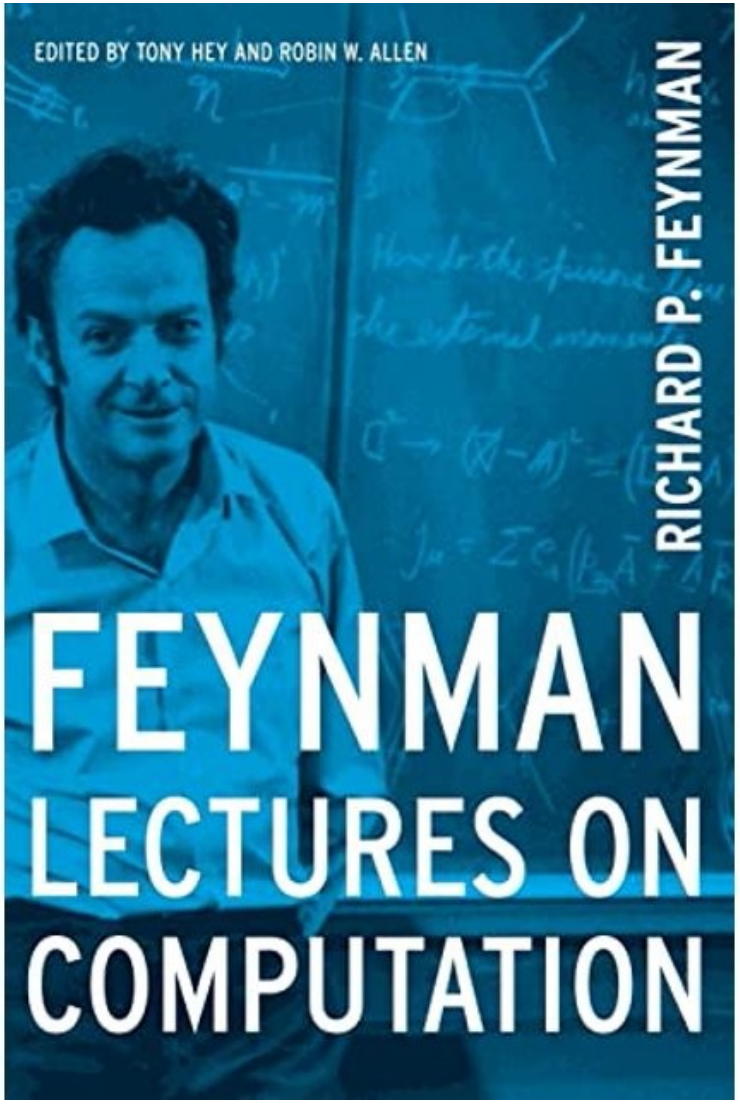
From… https://jedcrandall.github.io/usenix10.pdf

FOCI 2014 Knockel *et al.* slides...

USENIX Securtiy 2016 Cao *et al.* Slides...

References

- *NMAP NETWORK SCANNING*, by Gordon “Fyodor” Lyon
- Google “nmap”, “idle scan”, etc.
- Other references were linked to inline


EDITED BY TONY HEY AND ROBIN W. ALLEN
RICHARD P. FEYNMAN
FEYNMAN
LECTURES ON
COMPUTATION