Denial of Service Defense in Practice and Theory

Eddie Kohler
UCLA/Mazu Networks

USENIX
April 13, 2005

About this talk

- Idiosyncratic
- Broad
- Shallow (±)

About the presenter

- Operating systems researcher
- Network protocol designer
- DDoS solution vendor (±)
- Panglossian
- Speaking solely for myself

What is denial of service?

- Resource exhaustion
- Attacker makes target resource unavailable to others
- Two victims: target resource, legitimate users

DoS characteristics

- Attacker gains intangible
  - Not like credit card theft, Web site defacing
- Attack can use innocent traffic or evil traffic
  - Malignant traffic: crash destination host
  - Pseudobenign traffic: take up resources (slow down destination)
- Theoretically impossible to distinguish DoS from legitimate traffic (“flash crowds”)

What causes denial of service?

- Wasted or useless work
- A program does work that is eventually thrown away
- Broad definition

  Congestion collapse is a DoS scenario

What resources are exhausted?

- Network bandwidth
- CPU
- File descriptors
- Server memory
- . . .

Distributed denial of service

- Many attackers, one victim
  - Attackers use *zombies*: compromised servers or Windows boxes
  - Or *source address spoofing*: appear to be many sources
  - The Dept. of Defense worries about national cyberwarfare
- Prototypical attacks: February 2000, Yahoo, Amazon, Ebay, . . .
  - Sites off the net for hours
  - $1.2B in damages (Yankee Group) (?!)
  - A thousand mitigation companies bloom (well, three)

The original DDoS attack

- ‘On April 15, everyone in China is going to jump up and down simultaneously at noon, knocking the earth off its axis!’

The new DDoS attack

- ‘On April 15, everyone in China is going to download whitehouse.gov simultaneously at noon, knocking the government’s Web site off its axis!’

And yet. . .

- Incentives are changing
- In 2000, it was mafiaboy: a 15-year-old Canadian hacker who hung out bragging on IRC
- In 2005, it's the Russian mafia

The shadow economy

- Extortion
  - Online gambling
  - E-porn
  - Small-to-medium sites whose travails may not bother their service providers
- Symbiotic world of malware
  - Break into a machine with a worm, sell access for spam/DDoS
  - Spam proxying: 3–10¢/host/week
  - Millions of hosts for sale

Preliminary conclusions

- DoS is here to stay (controversial, huh?)
- Arms race: no obvious winner, no obvious trend
- Good partial solutions available
- Solution choice motivated by several factors
    - Cost of false positives
    - Interactivity
- Need new operating systems
- Threat to small sites requires an architectural solution

Characteristics of DoS

- Malignant traffic
  - A relatively small number of packets can bring down infrastructure
  - Example: Christmas tree packets, ping of death
  - Cause is endemic computer engineer disease: insufficient consideration of error cases
- Pseudobenign traffic
  - Any individual packet's OK, only the volume of requests matters
  - Problematic volume depends on work induced by packet
  - Examples: smurf, SYN flood

Complicating factors

- Reflection
- Amplification
- Attack through defense

Reflection & amplification

- Attacker tricks a third party into attacking
  - Particularly bad if third party sends more traffic than attacker: *amplification*
- Canonical example: smurf
  - Send ping to IP local broadcast address
  - Spoofed source address = target
  - Result: a whole network replies to the target
- DNS vulnerable even without spoofed source address
  - Recursive lookups: "look up X, tell Y answer"
  - Look up something huge (DNSSEC)

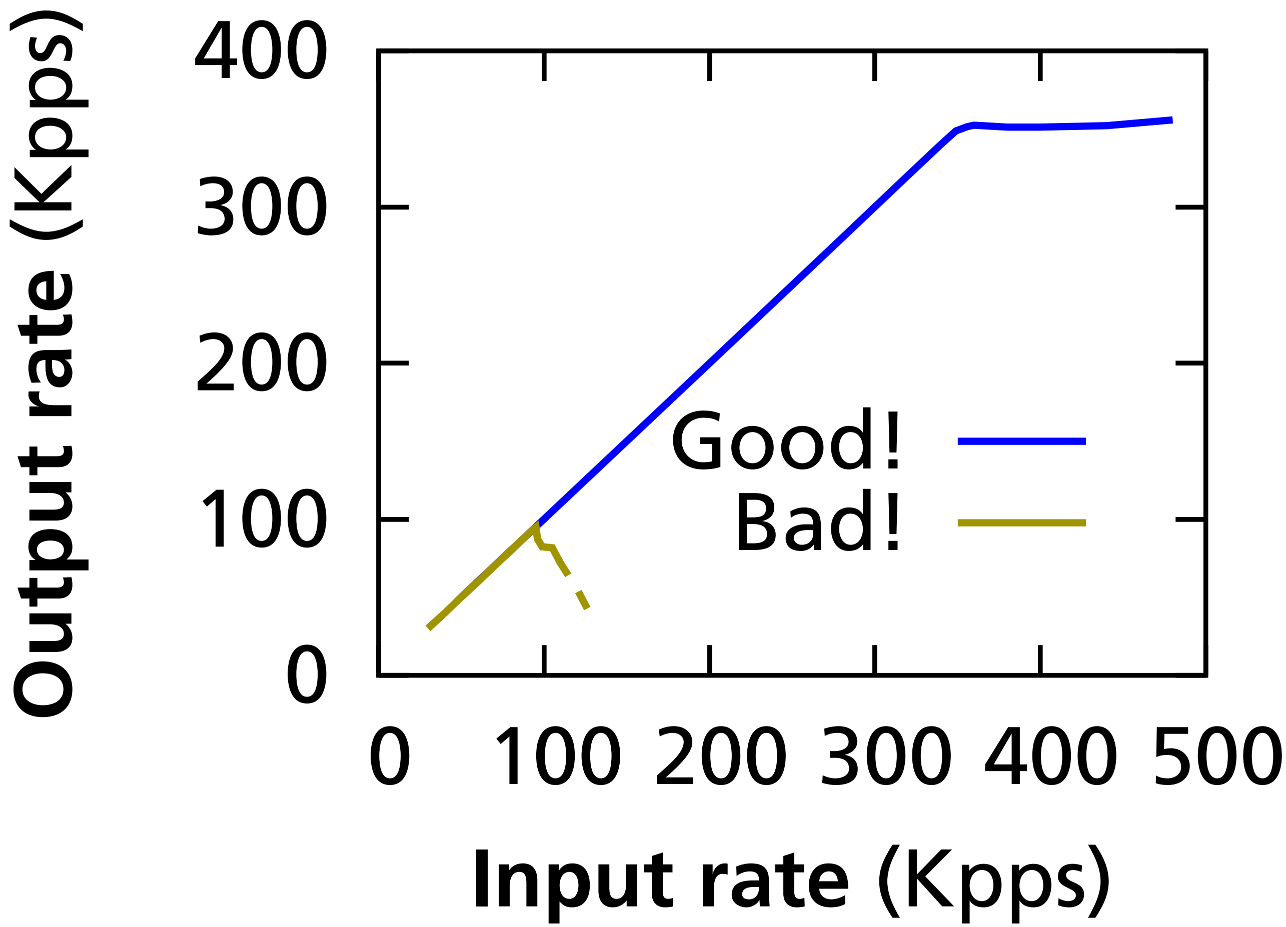
Attack through defense

- Attacker chooses victim
- Tricks network defense mechanism into treating victim as attacker
- Use intelligent network against itself
- Relies on source address spoofing or traffic aggregation

DoS solution classes

- Ensure any work is meaningful
    - Authentication and encryption
    - Drop work as early as possible
- Offload work
    - Servers considered vulnerable
    - Force clients to do the work
- Identify attackers
    - Sounds impossible, is not

Two performance curves



Receive livelock as DoS opportunity

- Interrupt-driven network I/O
- One interrupt per packet arrival
- Interrupt gets priority over all other system processing
  - *Including other arrived packets*
- Result: System reduces to handling only interrupts
- Wasted work

Solution: Polling

- Don't waste work
- Prioritize partial effort over no effort
- Drop work **early**
- Polling: Ask cards for packets
  - Puts CPU in charge of relative prioritization
  - Packets are dropped on the input card
- Linux NAPI, FreeBSD polling

Connection state

- TCP Transmission Control Blocks

  Connection state, sequence numbers

- Receive SYN, create TCB

  Need to verify ACK against existing connection

- Classic DoS attack: SYN flood
- Send SYNs with fake sources
- Victim responds
- Takes up connection state until timeout

Digression: Faked sources or not?

- 2000 conventional wisdom: Spoofing is a disaster
  - Egress filtering (don’t emit packet you wouldn’t accept)
  - IP traceback

IP traceback

- Goal: Destinations can infer any packet's full router path
- Query routers about particular packets?
- Routers store path in IP option?
- Routers probabilistically encode path segments in IP ID?

    Need many packets to reconstruct

2005 conventional wisdom

- Fake? Real? Doesn't matter
- Sources are always zombies anyway
- Assume all sources are real
- In fact, we still observe many faked-source attacks

SYN flood response

- Reduce state
- Smaller TCB for SYN-RECEIVED connections
- SYN queue
    - Keep queue of SYN-RECEIVED connections
    - On ACK of SYNACK ($\rightarrow$ ESTABLISHED), remove connection from queue
    - Under attack, queue will overflow
    - Throw out oldest unacked connection
- Remote SYN queue
    - Offload SYN queue from host onto middlebox
    - Send RST on overflow

Better SYN flood response

- **SYN cookies**
- On SYN, encode all connection information in cryptographic cookie
  - → Sequence number
- On ACK for unknown connection, check cookie
  - If invalid, drop/send RST
  - If valid, instantiate TCB

The principle

- Offload state

  Wasted state is wasted work

- TCP is lucky: sequence number is enough for cookies
- What if your protocol has more information?
- Add an explicit cookie
- Cookie offloads state to client
- Client must echo cookie to server

Cookie risk

- Example cookie and more: TCP-MD5
- MD5-sum every packet
- Cheap-ass authentication
- Still requires MD5 check on every packet
- Attacker can induce work by sending bogus MD5sums – you must check them!
- Cryptography $\Longrightarrow$ denial-of-service
- Checking an invalid hash/signature is wasted work
- Need to minimize
- Sequence number security has real advantages!

Minimize work by doing more work

- Example: CNN, 9/11
- DDoS made up of real users who wanted real data
- Solution: Put the entire CNN homepage in a single packet
  - Redesign content
  - Whole hog: No TCBs; send a SYNACK for every SYN, a data packet + FIN for every ACK
- Blocking the attacker isn't always the solution
  - Make it cheaper to respond to everyone

On "attack mode"

- Behave normally when not under attack, conservatively when under attack
- Tempting idea, suggested again and again
- Problem: usually involves *doing more work when under attack*
- Also easy to attack-through-defense

Kill-Bots

- Interesting attack-mode algorithm (Katabi et al, NSDI 05)
- When under attack, introduce puzzle people must solve to continue
    - Ticketmaster-style
- Puzzle fits in a packet – cheap
- Puzzle response cheap to check

TCP RST attacks

- Send a packet, reset a connection
- TCP accepts any RST in the window
- Clearly DoS
- Response: shrink window for RSTs
- OK, but what about SYNs?
- Only authenticated packets should cause actions that can kill the connection

Protocol recommendations

- Sequence number security

  Big sequence numbers

  If you must use small sequence numbers, don’t allow connection close
- Big cookies
- Rate limits

  Allow protocol to degrade smoothly when host is too busy

  For example, don’t send a RST for an out-of-sequence packet

Identifying sources

- Reverse Turing test
- Done at speed
- Millions of packets a second
- Impossible
- Does that matter?

Attack classification

- Big
    - Bandwidth
    - Big site
    - National attack
    - ISP operators notice
- Medium
    - Server host resources
    - Data center network resources
    - Smaller site/more localized attack
    - ISP operators might not notice

Defense classification

- Big
  - Collateral damage/false positives not a big issue
  - Everything's horrible anyway
  - Solve it in the ISP
- Medium
  - Collateral damage/false positives larger issue
  - ISP cares less/has less leverage
  - That's OK, it's small: address it at least partially at the edge

Solving big attacks

- Currently, “solving it in the ISP” is a horrible manual process
- Hours-long phone calls
- Inter-ISP trust issues
- Can’t extract information from routers

  Turn on NetFlow, watch MLFFR drop

- Want to automate
- Pushback
- Networks of monitors

Pushback/Aggregate-based congestion control

- Router sends congestion signal
- Router examines traffic, finds *aggregate* that isn't responding to congestion signal, filters
- *Push congestion back* towards the source
- Eventually hits compromised router, but minimize wasted work for the rest of the network
- How to ensure pushback comes from a proper source?

  TTL hack

Network of monitors

- Boxes look for attack
- Communicate limited information among ISPs
- Coordinate response
- Automate response . . . ?

Edge mitigation

- No magic bullet – or, rather, many magic bullets
- **Visibility**
- Automated filter construction
- Based on traffic baselines
- Based on attack trajectory
- Based on user input
- Data structure design

  Unit work no matter the traffic conditions
- $\Longrightarrow$ Mazu

Examples

More dimensions

- Applications
- Server structure

  Minimize per-connection usage: event-driven servers

- Host resources
- Cleverer attackers

  Home page downloads the current cutting edge

- Architectural solutions

  Can’t even name something you’re not authorized to talk to

Thoughts in lieu of conclusion

- Beware of rearchitecting the network

  Remember the CNN lesson: More work for less cost

  New architectures

- Beware of false positives

  You *want* to keep attacks in the network

  Until they reach the narrow waist where they crowd out legitimate traffic

  (Depending on cost structure of course)

- Rearchitect OS if anything