# Foundations of Network and Computer Security

**J**ohn Black

Lecture #14
Oct 11th 2004

CSCI 6268/TLEN 5831, Fall 2004

# Announcements

Midterm Thursday

- Material: lectures through today; all readings; all projects (not silly OpenSSL details)
- Closed notes; calculators allowed
- Exam is 4 pages long (double-sided)
  - 2.5 pages short-answer
  - 0.5 pages extended topic
  - 1 page of justified true/false
- Half of exam is regurgitation, half is thought-problems
  - Some thought problems are hard
    - Best strategy: do easiest problems first
    - For hard problems, if you can't solve it, prove you at least know what the goal is (for partial credit)

# Colloquium Talk

- Thursday, Oct 14th, 3:30pm, ECCR 265
- This is my “reappointment talk”
- Might be kind of redundant for people in this class, but come if you have nothing better to do

# Tracebacks Methods

- One basic problem with fighting DDoS is that we cannot find the *source IP* of the attacker
  - Finding the attacker would allow us to shut down the attack at the source
    - This assumes ISPs will cooperate and that there is a mechanism in place for reporting the source
    - Both of these assumptions are questionable as we saw in the Gibson story
  - The Internet Protocol (IP) makes it hard to find out where things are coming from
    - Easy to forge source IPs
    - No tracing mechanism available
      - This is on purpose

# Adding Traceback

- Perhaps we could add a mechanism to IP to implement traceback
  - Still doesn't stop reflectors
  - Needs to be backward-compatible with current routing protocols
    - If not, too expensive and no one will do it
  - There have been several suggestions
    - Probabilistic traceback
    - Algebraic traceback
    - Others
  - We'll look just at probabilistic traceback

# Probabilistic Traceback

- Original idea due to Savage, Wetherall, Karlin, and Anderson
  - "Practical Network Support for IP Traceback"
- Improved scheme due to Song and Perrig
  - "Advanced and Authenticated Marking Schemes for IP Traceback"
- We'll focus on the first paper, even though it is still far from a complete solution

# First Try: Link Testing

- Idea: Manually trace source of traffic
  - Too labor intensive
  - Some tools developed, but requires a lot of cooperation between ISPs and backbone companies
    - Not much economic incentive to cooperate
  - Could use “controlled flooding”
    - Induce traffic from upstream routers and see which traffic is dropped
    - But this is a DoS attack itself… ethical?
    - Relies on being able to generate traffic
    - Requires good map of the Internet… hard to get
  - Both are useful only *during* an attack

# How about Logging?

- Idea: select routers log all packets as they pass through
  - Then what?
  - Data mining techniques to try and figure out which packets were part of an attack
  - Then trace back upstream
  - Huge resource requirements!
  - Large-scale inter-provider database integration problems

# Packet Marking

- Idea: mark packets as they pass through routers
  - The mark should give information as to what route the packet took
  - One idea is to mark every packet that traverses a given router
    - Just append their IP address to a list in the IP header
    - Drawback is that this is a HUGE burden to put on routers
      - They would have to mark EVERY packet
      - Packets would get enormous if they travel a long route
      - Packets might be caused to fragment

# Probabilistic Packet Marking

- First, some assumptions:
  - Attackers can generate any packet
  - Attackers can conspire
  - Packets can be lost or reordered
  - Route from attacker to victim is mostly stable
  - Routers are not widely compromised

# PPM: Continued

- Each router writes its address in a 32-bit field only with probability p
  - Routers don't care if they are overwriting another router's address
  - Probability of seeing the mark of a router d hops away is $p(1-p)^{d-1}$
  - This is monotonic so victim can sort by number of packets received and get the path
    - Smallest number is received by furthest router, etc

# PPM: Difficulties

- We have to change the IP header any time a router marks a packet
  - This means storing the mark (has drawbacks)
  - Updating the header checksum
    - But this is already done for TTL decrements
- But we may need a LOT of packets to reconstruct a path
  - Suppose p=0.51 and d=15, then we need more than 42,000 to get a single sample from the furthest router
  - To get the order right with 95% probability requires around 300,000 packets
- Multiple attackers complicates matters
  - With multiple attackers at the same distance, this all breaks down

# Next Try: Edge Sampling

- Reserve two address-sized fields in the IP header: “start” and “end”
- Reserve a small “distance” field as well
- When a router decides to mark a packet, it writes its address in the “start” field and zeroes the distance field
- When a router sees a zero in the distance field, it writes its address in the “end” field
- If a router decides not to mark a packet, it increments the distance field only
  - Must use saturating addition
  - This is critical to minimize spoofing by the attacker; without it, attackers could inject routers close to the victim
  - Now attacker can only spoof marks with distance counts equal or greater than its distance from the victim
- Note that we can now use any probability p we like
  - We’re not sorting based on packet counts any longer

# Edge Sampling (Cont)

- The expected number of packets needed for the victim to reconstruct the entire path is at most $\ln(d)/p(1-p)^{d-1}$
  - Example: p=0.1, d=10, reconstruction requires about 75 packets
  - This is related to the coupon-collection problem
- Edge sampling allows reconstruction of the whole attack tree
- Encoding start, end, and distance is a problem
  - Not backward compatible if we change the IP header!
  - There are ways around this

# Digression: Coupon Collection

- Suppose you have t types of coupons, $C_1$, $C_2$, … , $C_t$
  - Each time you open baseball cards, you get a coupon of type i with probability 1/t
  - How many coupons do you need before you have a complete set?
    - Note that in real competitions, all types are not 1/t
    - Call total number you need N (a random variable)
  - Define a random variable $N_i$ indicating the number of draws you need to use when you hold i-1 coupon types and you want a new type
    - Then $N = N_1 + N_2 + \ldots + N_t$

# Coupon Collection (cont)

- Then $E(N) = E(N_1) + \ldots E(N_t)$
  - Linearity of expectation
- What is $E(N_i)$?
  - Probability of getting a new type if you have i-1 types is (N-i+1)/N, so expectation is N/(N-i+1)
    - For geometric random variables, expectation is the inverse of the parameter
    - If you have a fair die, it takes an expected 6 rolls to get a 4 (for example)
  - So $E(N) = 1 + N/(N-2) + N/(N-3) + \ldots + N$ or $N H_N$
    - Here $H_N$ is the Nth harmonic number
    - This is approximately N ln N.

# Back to Reality

- No one does this
  - Yet?!
- DDoS attacks are still a huge problem and are still quite common
- But fortunately there is even more to worry about

# TCP Session Hijacking

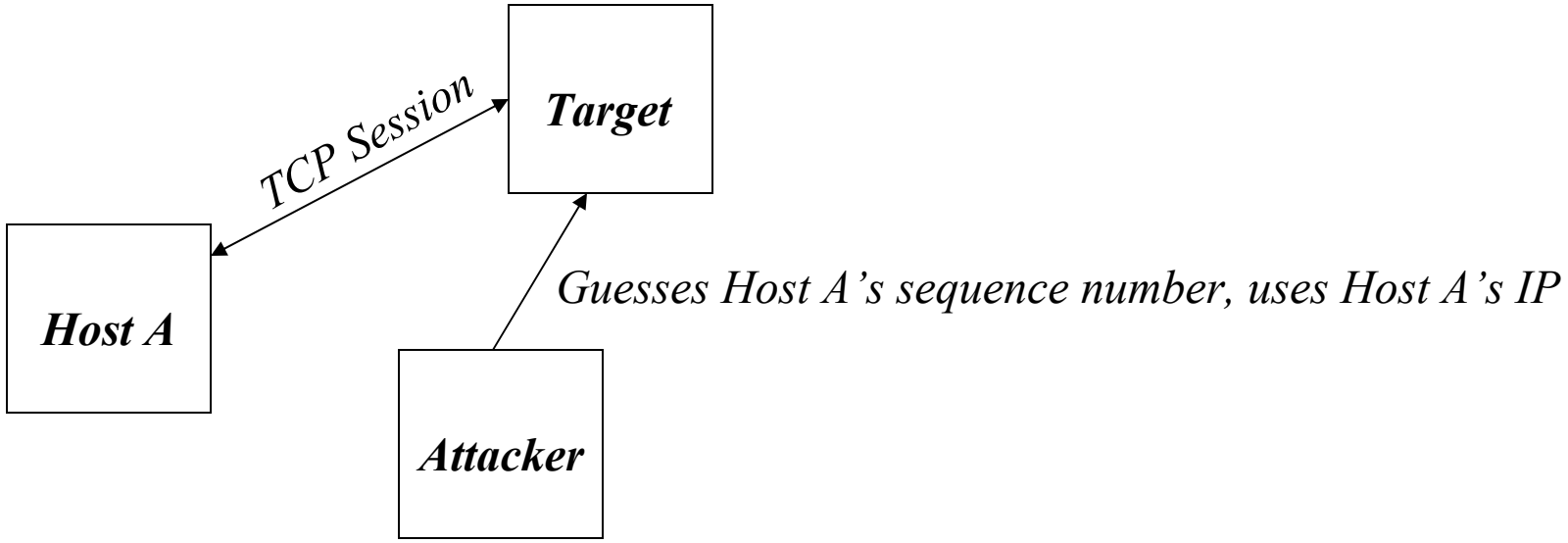
- This is the last topic on network-based attacks for a while
- After the midterm we'll look at vulnerabilities for awhile
- We'll come back to some network protocols and some more crypto later in the course

# Session Hijacking

- How might we jump in on an established TCP session?
  - If we could sniff the connections and inject traffic, we could do this with no problem
  - If we can only inject traffic (by sending unsolicited TCP segments to the victim) it's harder
    - Must guess the proper sequence number

# Hijacking

- If attacker uses sequence number outside the window of Target, Target will drop traffic
- If attacker uses sequence number within window, Target accepts as from Host A
  - Result is a one-sided connection
  - Can be used to crash Target, confuse, reset connection, etc



# Preventing Hijacking

- Make sequence number hard-to-guess
  - Use random ISNs
    - Note that SYN cookies in effect do this by using a hash of some stuff which includes a counter and a secret key
- There are many other kinds of hijacking techniques
  - We'll later look at ARP cache poisoning