Network Attacks Review & Denial-of-Service (DoS)

CS 161: Computer Security

Prof. Vern Paxson

TAs: Devdatta Akhawe, Mobin Javed & Matthias Vallentin

http://inst.eecs.berkeley.edu/~cs161/

February 15, 2011

Goals For Today

- Review the different classes of network attacks and how they relate to network *layering*
  - Feedback requested: was this valuable?

- Discuss Denial-of-Service (DoS): attacks on *availability*
  - Mostly network-based, but also OS

Basic Types of Security Goals

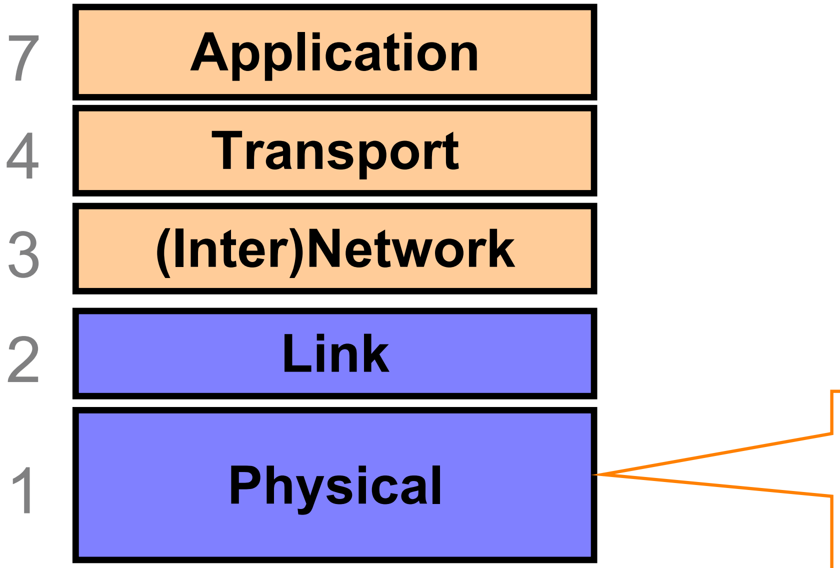
- Confidentiality:
  - No one can *read* our data / communication unless we want them to
- Integrity
  - No one can *manipulate* our data / processing / communication unless we want them to
- Availability
  - We can *access* our data / conduct our processing / use our communication capabilities when we want to

Types of Security Goals, con't

- Attacks can subvert each type of goal
  - Confidentiality: eavesdropping / theft of information
  - Integrity: altering data, manipulating execution (e.g., code injection)
  - Availability: *denial-of-service*

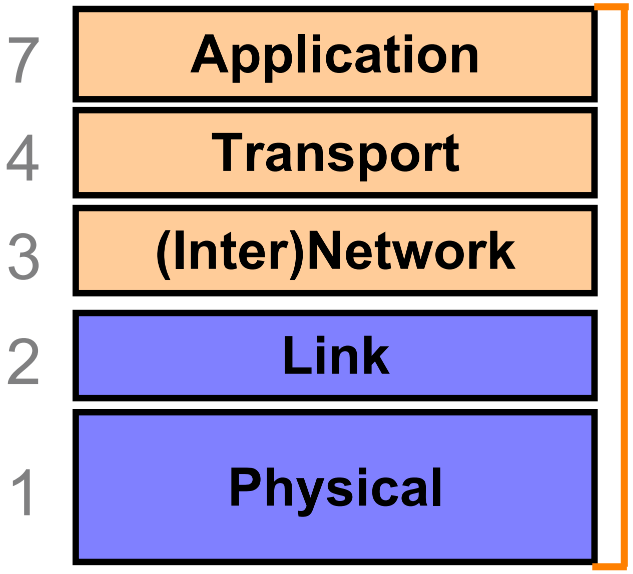
- Attackers can also combine different types of attacks towards an overarching goal
  - E.g. use eavesdropping (*confidentiality*) to construct a spoofing attack (*integrity*) that tells a server to drop an important connection (*availability*)

Network Attacks on Confidentiality



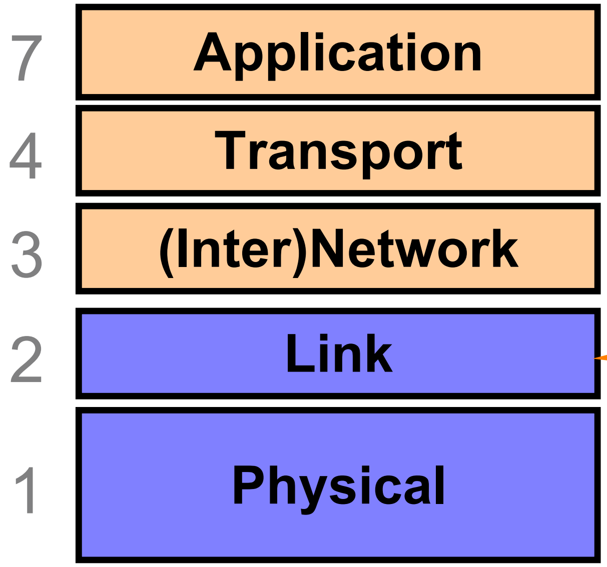
Nature of physical signaling can allow eavesdropping by nearby attackers

Network Attacks on Confidentiality



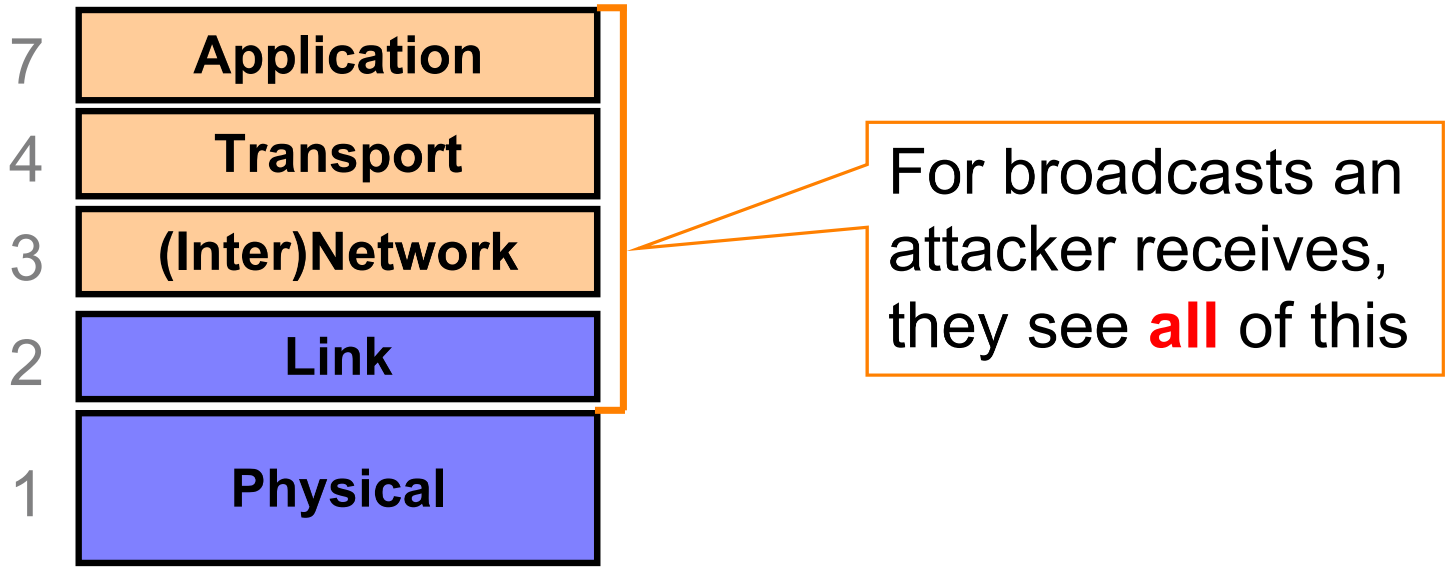
If they can eavesdrop, they see **all** of this

Network Attacks on Confidentiality



Some link layers (e.g., wired Ethernet) also allow attackers to receive subnet traffic sent w/ broadcast (such as DHCP)

Network Attacks on Confidentiality



7 Application

4 Transport

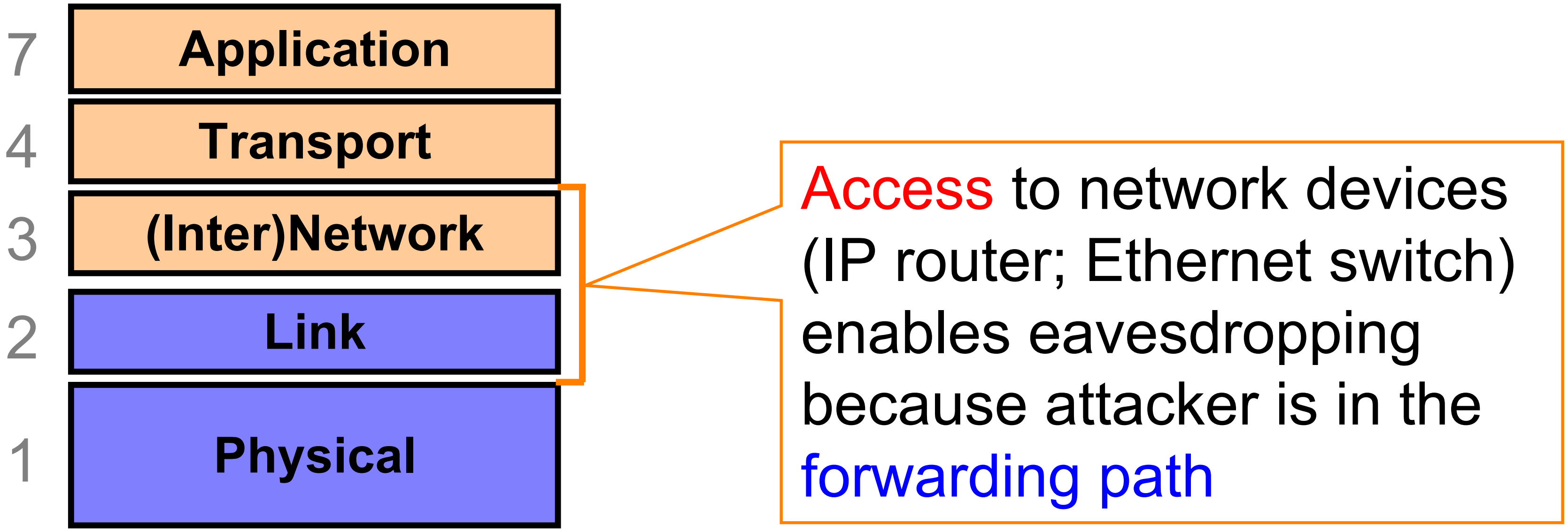
3 (Inter)Network

2 Link

1 Physical

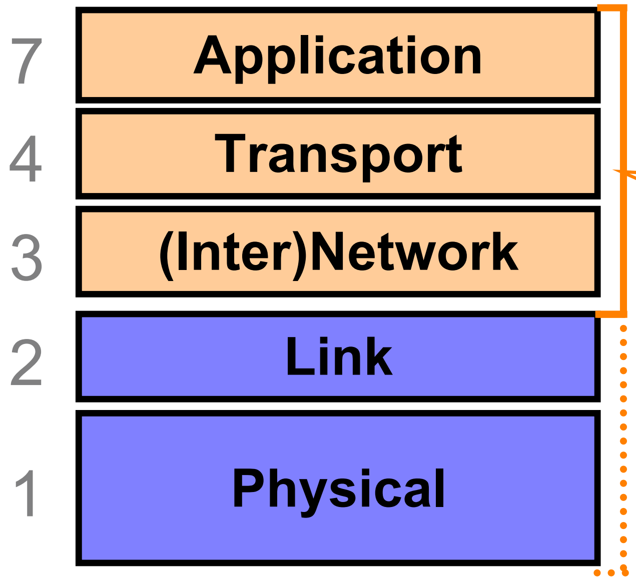
For broadcasts an attacker receives, they see **all** of this

Network Attacks on Confidentiality



Access to network devices (IP router; Ethernet switch) enables eavesdropping because attacker is in the forwarding path

Network Attacks on Confidentiality



7 Application

4 Transport

3 (Inter)Network

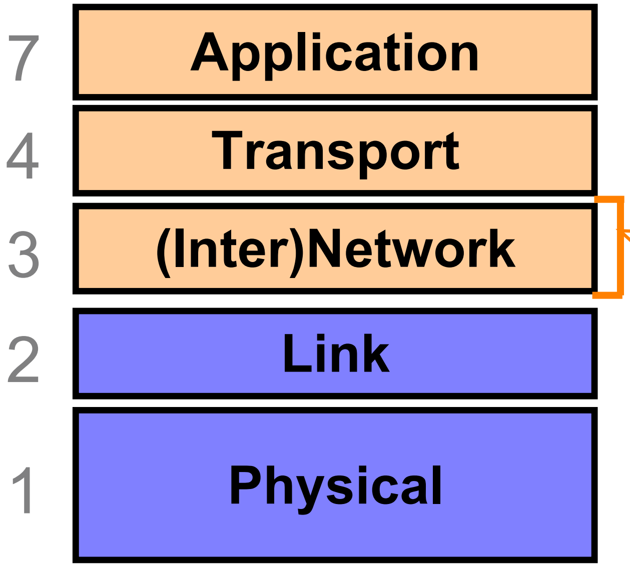
2 Link

1 Physical

If an attacker is in the forwarding path, they see **all** of layers 3/4/7 …

… and perhaps layers 1 and 2 too, depending on their location

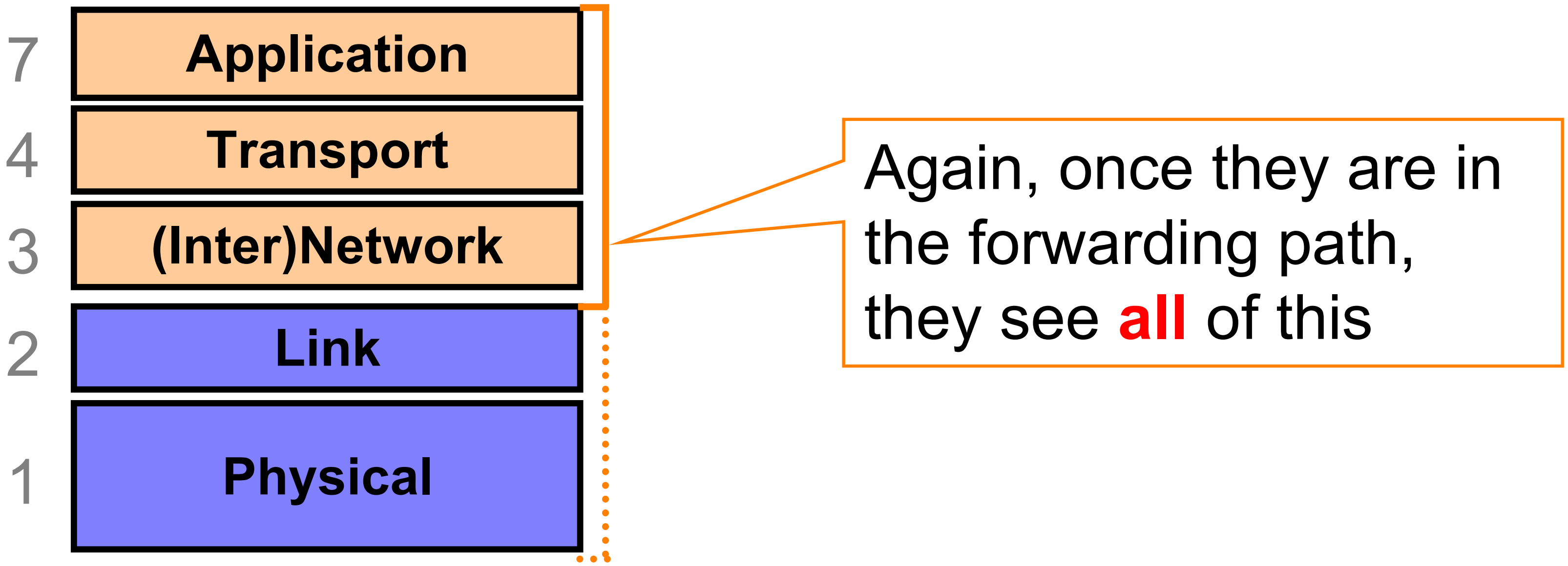
Network Attacks on Confidentiality



Attackers can insert themselves into the forwarding path if they can manipulate victims to send their traffic through systems controlled by the attacker

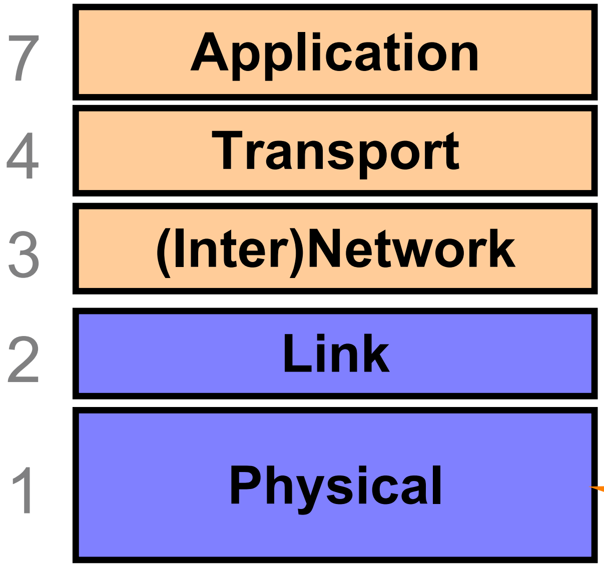
(E.g., DHCP spoofing to alter “gateway”, or DNS cache poisoning to alter a server’s IP address)

Network Attacks on Confidentiality



Again, once they are in the forwarding path, they see **all** of this

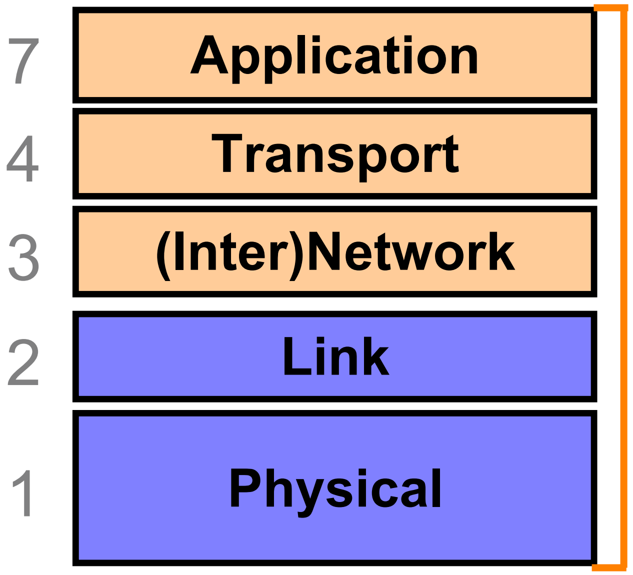
Network Attacks on Integrity



Access to ANY network allows attacker to spoof packets.

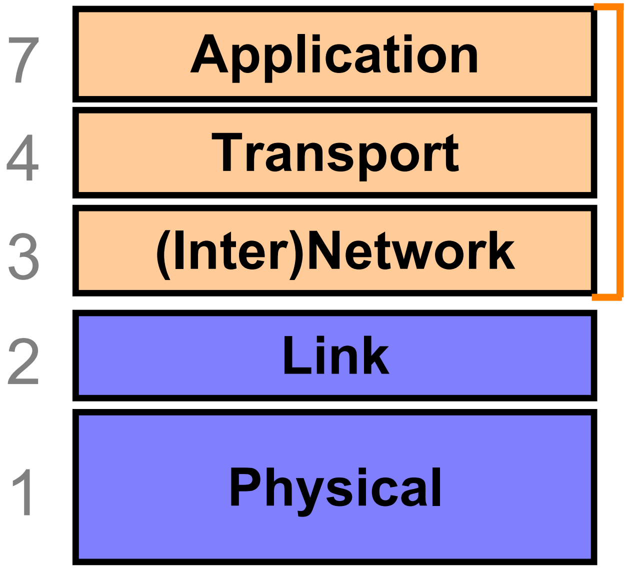
Spoof = send packets that claim to be from someone else.

Network Attacks on Integrity



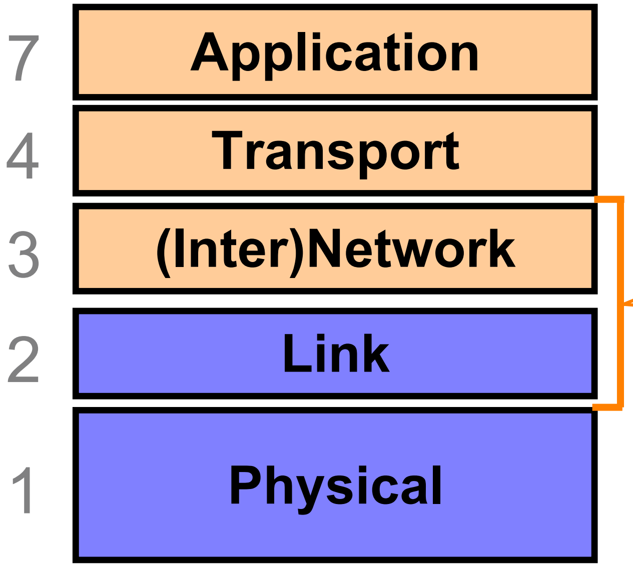
Once they can spoof, they can falsify **any/all** of this

Network Attacks on Integrity



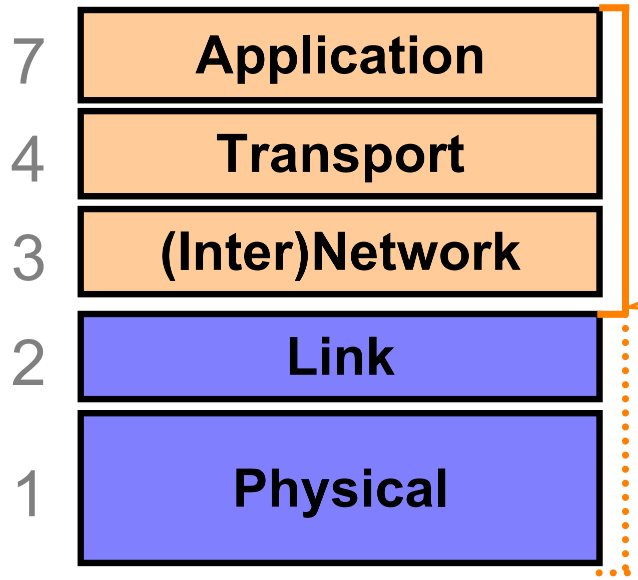
(… or if the NIC lacks programmability, then these)

Network Attacks on Integrity



Similarly, attackers who can get themselves on the forwarding path …

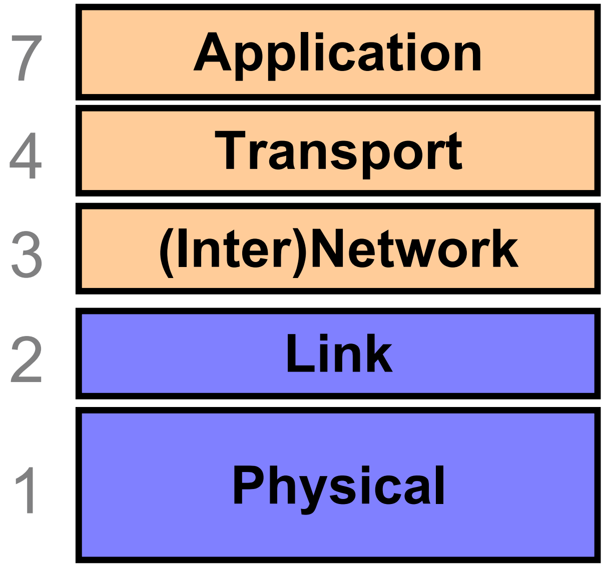
Network Attacks on Integrity



Similarly, attackers who can get themselves on the forwarding path … can create ***or alter*** any/all of this

Man-in-the-Middle (MITM)

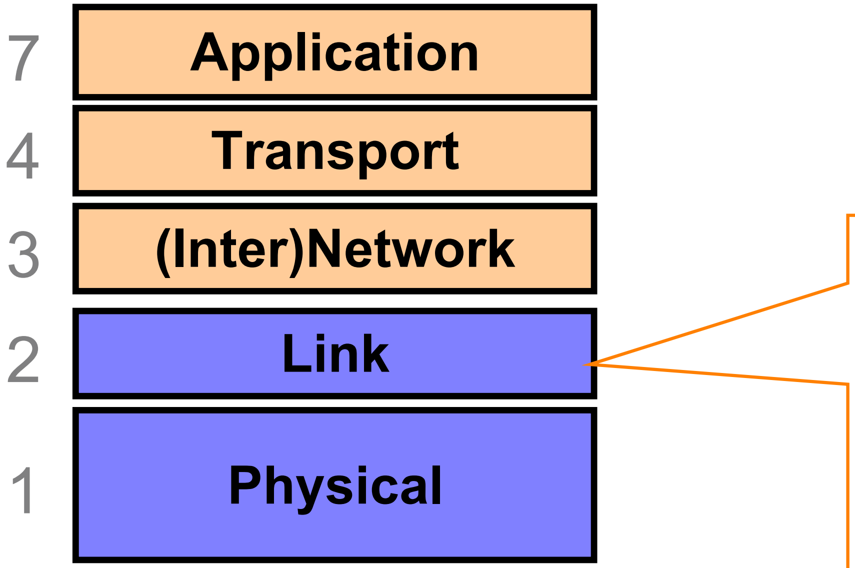
Combining Eavesdropping with Spoofing



To fool a receiver into accepting spoofed traffic, an attacker must supply correct Layer 2/3/4/7 values.

The easiest way to do so is to eavesdrop in order to discover the correct values to use.

Example: DHCP Spoofing



7 Application

4 Transport

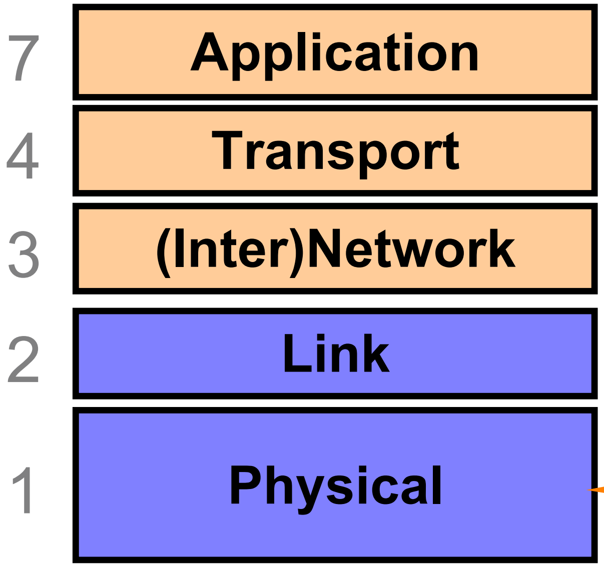
3 (Inter)Network

2 Link

1 Physical

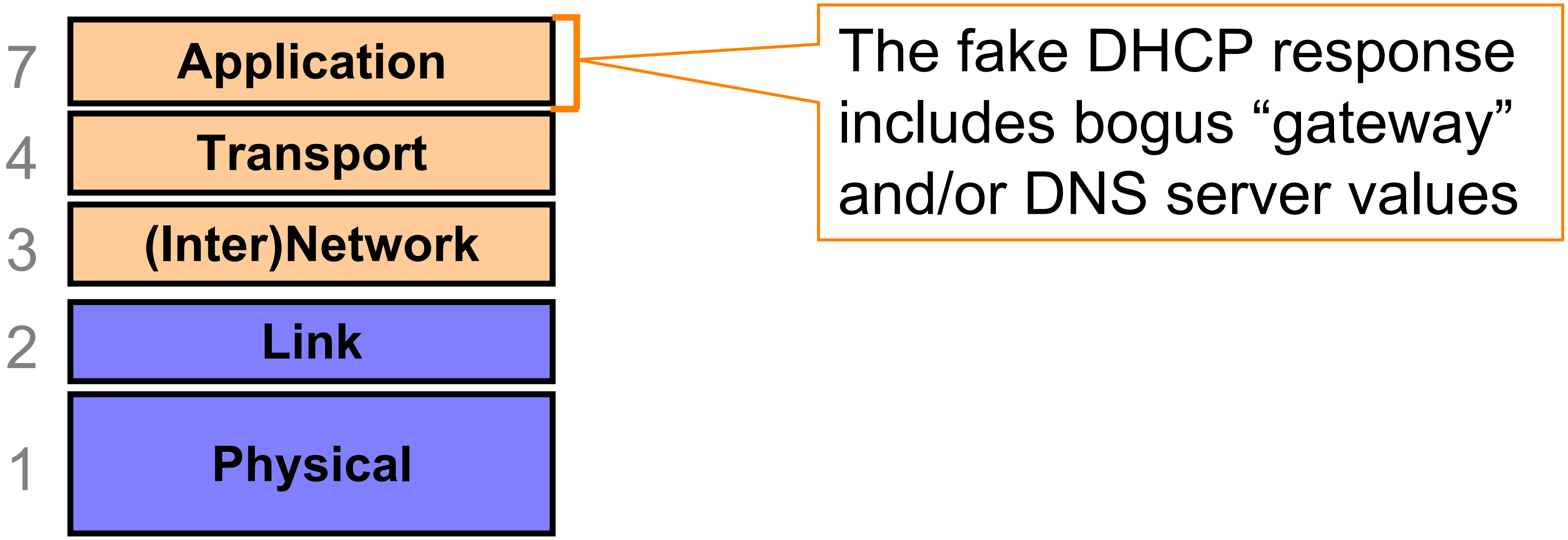
Attacker exploits link layer's broadcasting of DHCP requests to know when a client has a particular pending request

Example: DHCP Spoofing



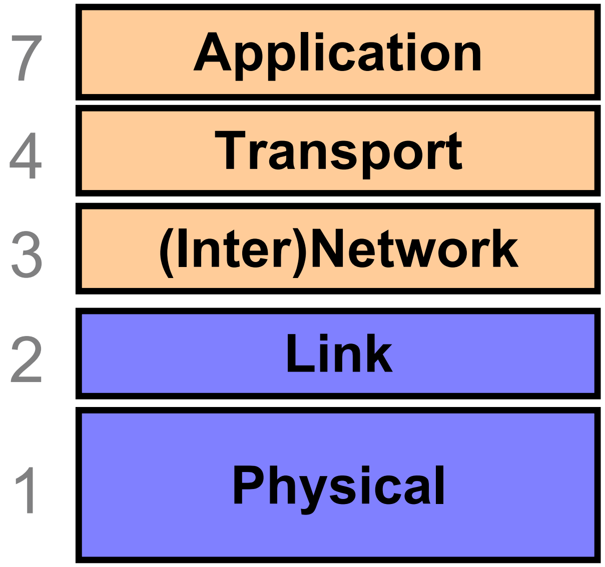
Attacker uses their direct access to network to spoof a corresponding DHCP response

Example: DHCP Spoofing



The fake DHCP response includes bogus “gateway” and/or DNS server values

Blind Spoofing



7 Application

4 Transport

3 (Inter)Network

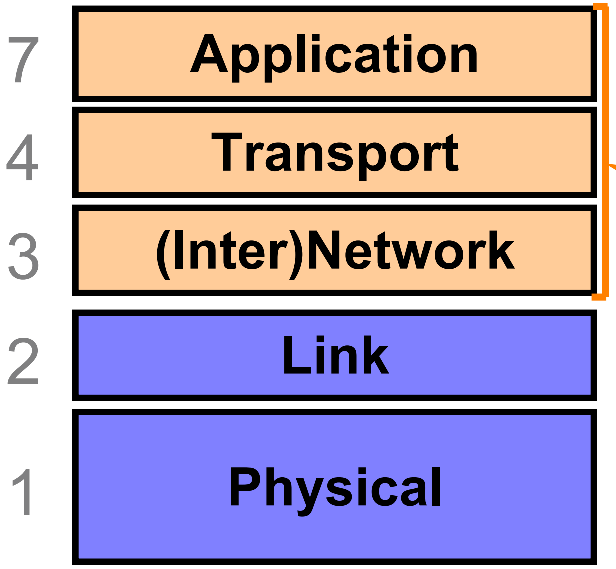
2 Link

1 Physical

To fool a receiver into accepting spoofed traffic, an attacker must supply correct Layer 2/3/4/7 values.

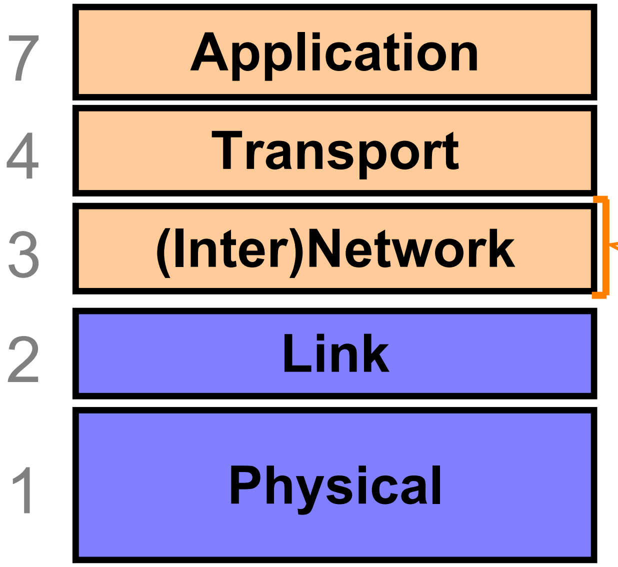
Another way to supply the correct values is to *guess*. Often requires additional information so “blind” guess has a prayer of being correct

Blind Spoofing



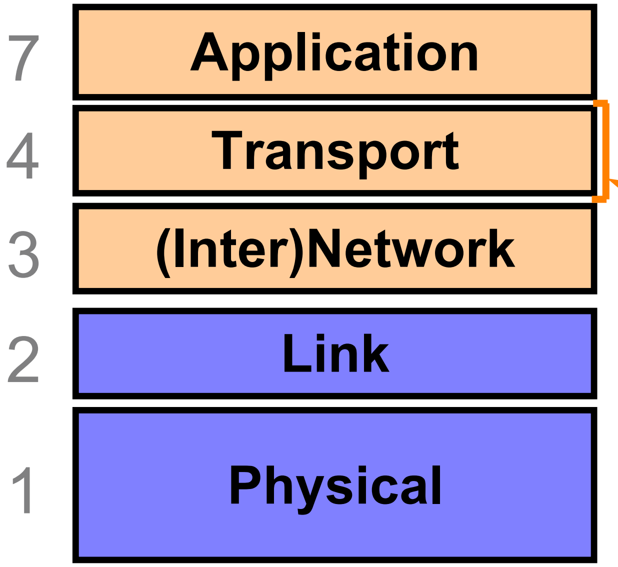
Remote attackers that can deduce layer 3/4/7 values can make receivers unwittingly accept unsolicited packets: ***blind spoofing***

Example: TCP Reset Injection



Attacker who can determine a connection’s IP addresses …

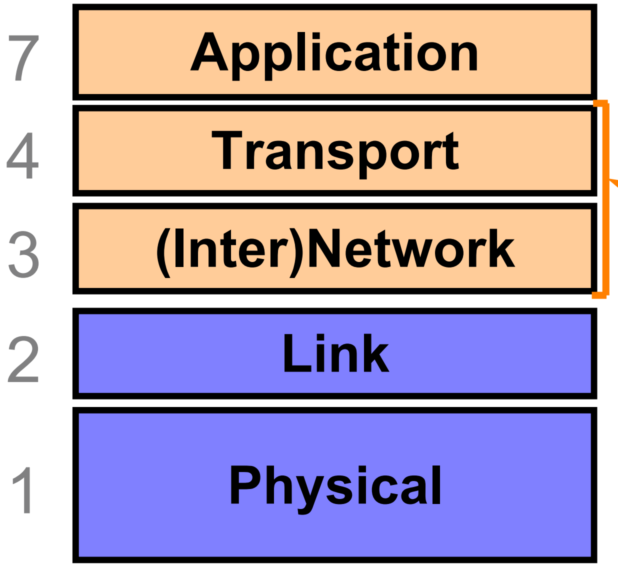
Example: TCP Reset Injection



Attacker who can determine a connection’s IP addresses …

… and TCP ports and sequence numbers …

Example: TCP Reset Injection

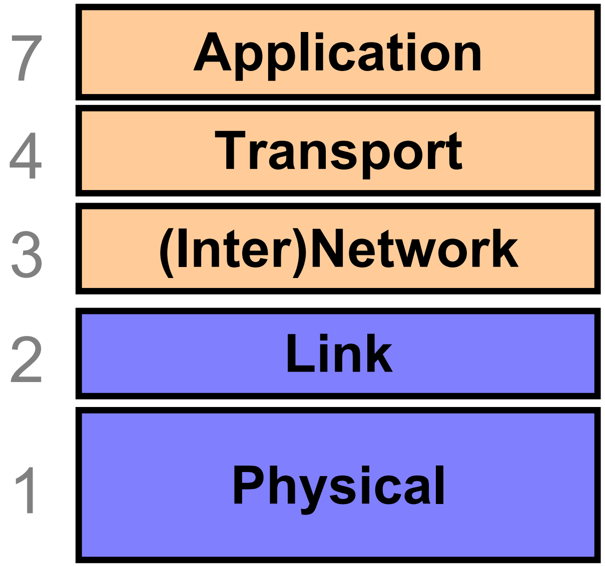


Attacker who can determine a connection's IP addresses …

… and TCP ports and sequence numbers …

… can forge a TCP packet with RST set that the receiver will be fooled into acting upon

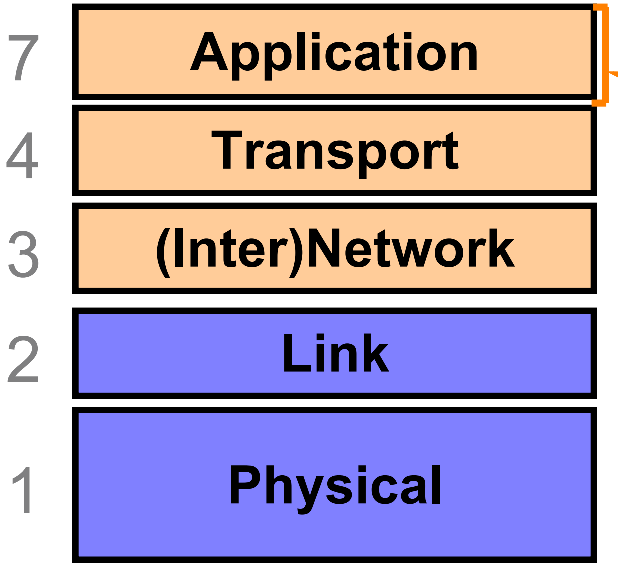
Violating Integrity Without Spoofing



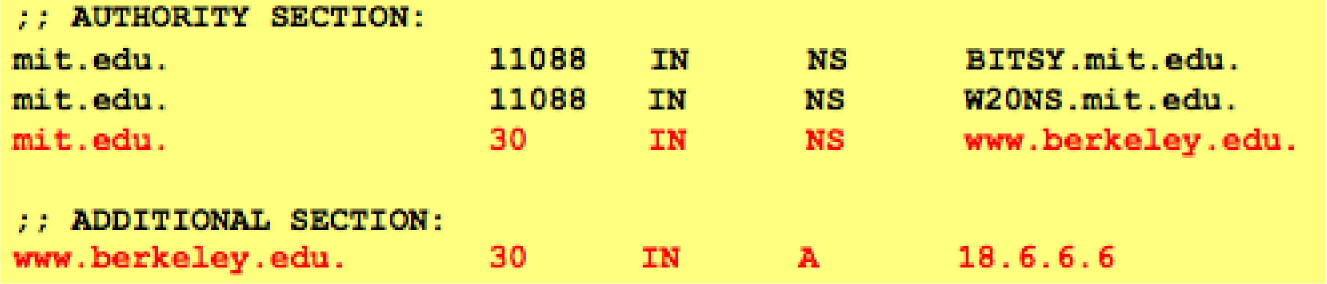
Depending on how an application protocol works, an attacker can directly manipulate its functioning …

… without *any* need to spoof.

Violating Integrity Without Spoofing

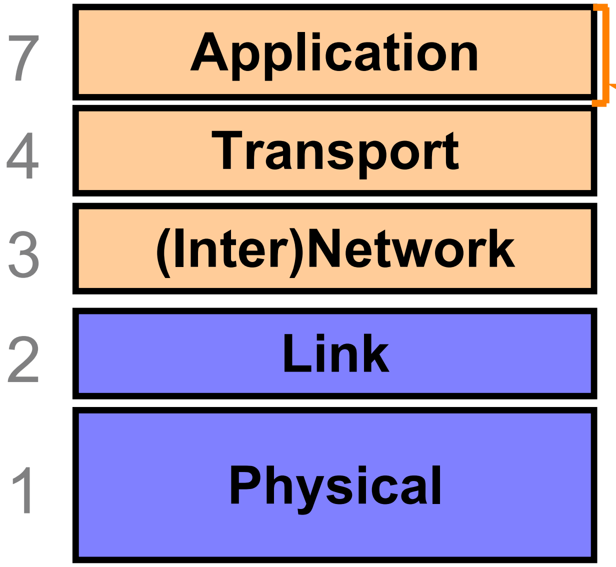


Our first example of DNS cache poisoning just involved an attacker manipulating layer-7 values.
No spoofing required.

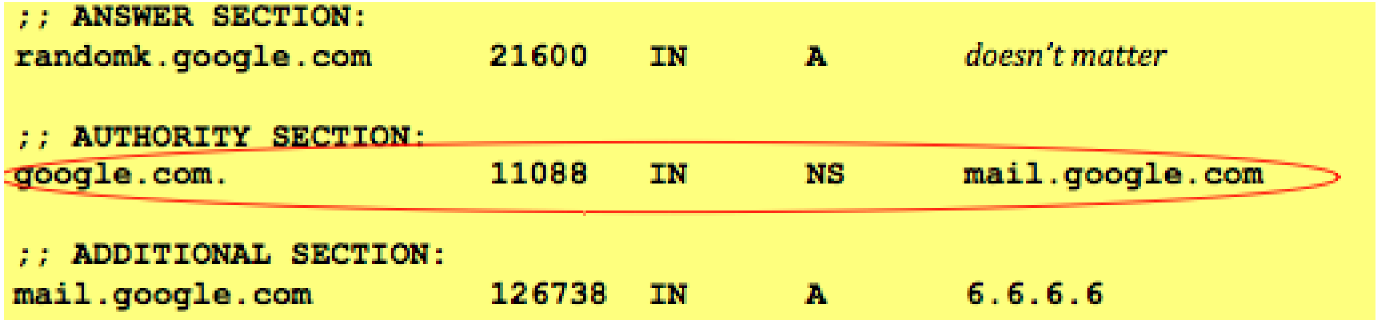


```
;; AUTHORITY SECTION:
mit.edu.                11088   IN      NS      BITSY.mit.edu.
mit.edu.                11088   IN      NS      W20NS.mit.edu.
mit.edu.                30      IN      NS      www.berkeley.edu.

;; ADDITIONAL SECTION:
www.berkeley.edu.       30      IN      A       18.6.6.6
```

Violating Integrity With Blind Spoofing

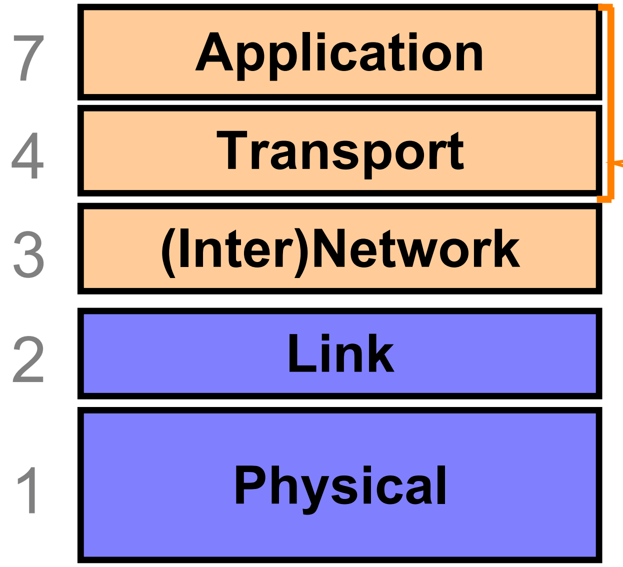


The *Kaminsky* attack, OTOH, repeatedly guesses the DNS transaction ID (layer 7), and sends traffic seemingly from the correct name server. Requires *blind spoofing*.

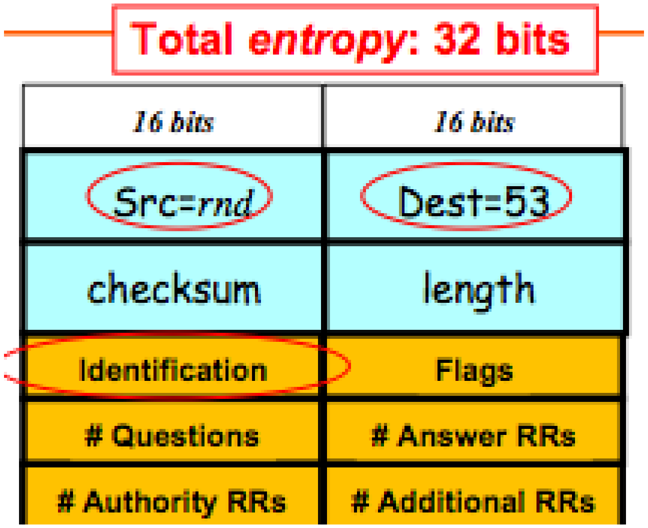


```
;; ANSWER SECTION:
randomk.google.com        21600    IN       A        doesn't matter

;; AUTHORITY SECTION:
google.com.               11088    IN       NS       mail.google.com

;; ADDITIONAL SECTION:
mail.google.com           126738   IN       A        6.6.6.6
```

Violating Integrity With Blind Spoofing



If we randomize the source port of our DNS requests, then attacker also has to guess a (16-bit) layer-4 value



Total *entropy*: 32 bits

16 bits	*16 bits*
Src=*rnd*	Dest=53
checksum	length
Identification	Flags
# Questions	# Answer RRs
# Authority RRs	# Additional RRs

5 Minute Break

Questions Before We Proceed?

Attacks on Availability

- Denial-of-Service (DoS, or "*doss*"): *keeping someone from using a computing service*
- Two basic approaches available to an attacker:
  - Deny service based on a program flaw
    - E.g., supply an input that crashes a server
    - E.g., fool a system into shutting down
  - Deny service based on resource exhaustion
    - E.g., consume CPU, memory, disk, network
- How broad is this sort of threat?
  - *Very*: **huge** attack surface
- We do though need to consider our threat model …
  - What might motivate a DoS attack?

Motivations for DoS

- Showing off / entertainment / ego
- Competitive advantage
  - Maybe commercial, maybe just to win
- Vendetta / denial-of-money
- Extortion
- Political statements
- Impair defenses
- Espionage
- Warfare

DoS Defense in General Terms

- Defending against program flaws requires:
  - Careful *authentication*
    - Don’t obey shut-down orders from imposters
  - Careful coding/testing/review
  - Consideration of behavior of defense mechanisms
    - E.g. buffer overflow detector that when triggered halts execution to prevent code injection ⇒ denial-of-service
- Defending resources from exhaustion can be **really** hard. Requires:
  - *Isolation mechanisms*
    - Keep adversary’s consumption from affecting others
  - *Reliable identification* of different users
    - Know who the adversary is in the first place!

DoS & Operating Systems

- How could you DoS a multi-user Unix system on which you have a login?

  -
    ```
    # rm -rf /
    ```
    - (if you have root - but then just “halt” works well!)

  -
    ```
    char buf[1024];
    int f = open("/tmp/junk");
    while (1) write(f, buf, sizeof(buf));
    ```
    - Gobble up all the disk space!

  -
    ```
    while (1) fork();
    ```
    - Create a zillion processes!
  - Create zillions of files, keep opening, reading, writing, deleting
    - Thrash the disk
  - … doubtless many more
- Defenses?
  - Isolate users / impose quotas

DoS & Networks

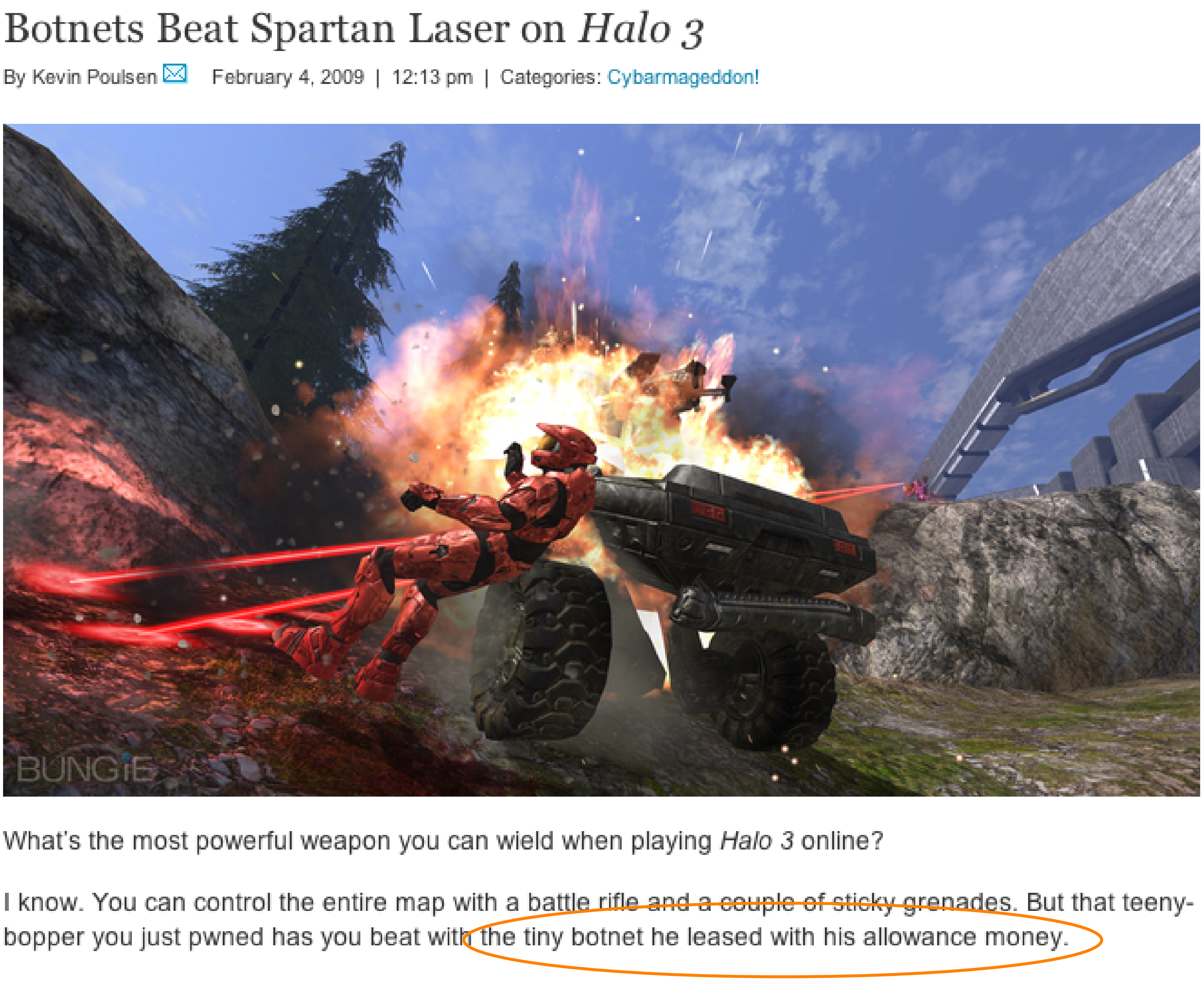
- How could you DoS a target’s Internet access?
  - Send a zillion packets at them
  - Internet lacks isolation between traffic of different users!
- What resources does attacker need to pull this off?
  - At least as much sending capacity (“bandwidth”) as the bottleneck link of the target’s Internet connection
    - Attacker sends maximum-sized packets
  - **Or**: overwhelm the rate at which the bottleneck router can process packets
    - Attacker sends minimum-sized packets! (in order to maximize the packet arrival rate)

Defending Against Network DoS

- Suppose an attacker has access to a beefy system with high-speed Internet access (a “big pipe”).
- They pump out packets towards the target at a very high rate.
- What might the target do to defend against the onslaught?
  - Install a network filter to discard any packets that arrive with attacker’s IP address as their source
    - E.g., `drop * 66.31.1.37:* -> *:*`
    - Or it can leverage *any other pattern* in the flooding traffic that’s not in benign traffic
  - Filter = *isolation mechanism*
  - Attacker’s IP address = means of *identifying* misbehaving user

Filtering Sounds Pretty Easy …

- … but it’s not. What steps can the attacker take to defeat the filtering?
  - Make traffic appear as though it’s from many hosts
    - Spoof the source address so it can’t be used to filter
      - Just pick a random 32-bit number of each packet sent
    - How does a defender filter this?
      - They don’t!
      - Best they can hope for is that operators around the world implement anti-spoofing mechanisms (today about 75% do)
  - Use many hosts to send traffic rather than just one
    - Distributed Denial-of-Service = **DDoS** (“dee-doss”)
    - Requires defender to install complex filters
    - How many hosts is “enough” for the attacker?
      - Today they are very cheap to acquire … :-(

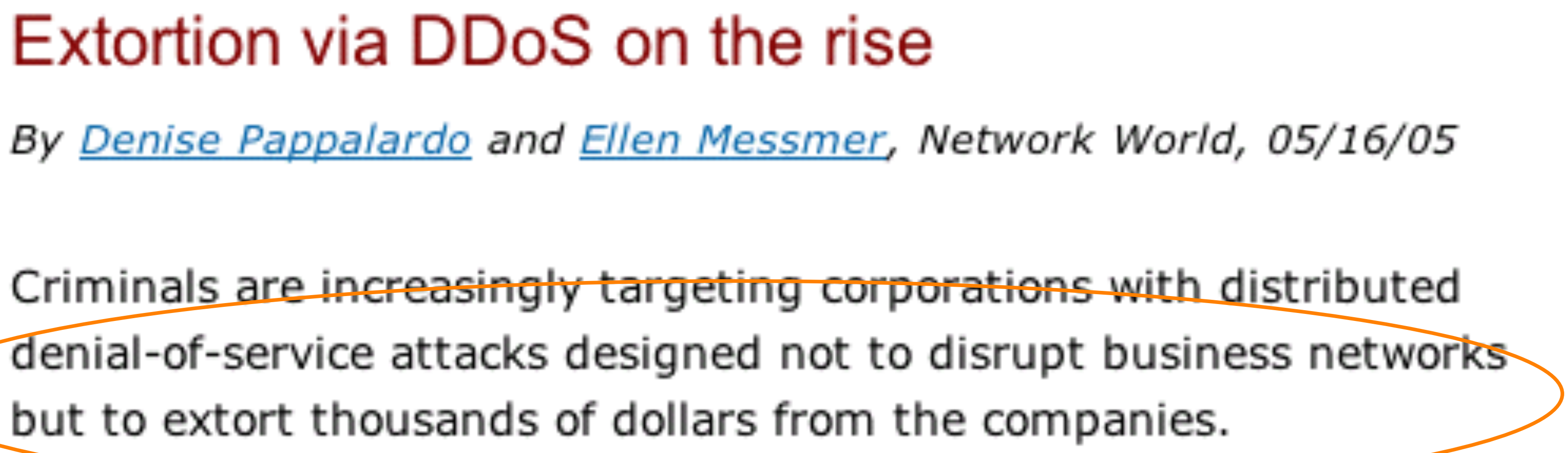


Botnets Beat Spartan Laser on *Halo 3*

By Kevin Poulsen ✉ February 4, 2009 | 12:13 pm | Categories: Cybarmageddon!

What's the most powerful weapon you can wield when playing *Halo 3* online?

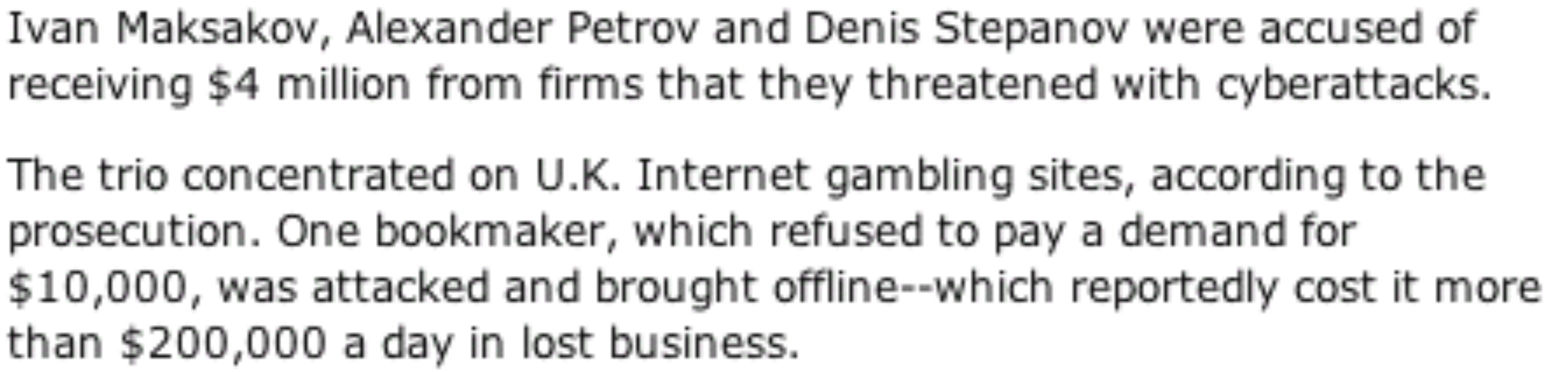
I know. You can control the entire map with a battle rifle and a couple of sticky grenades. But that teeny-bopper you just pwned has you beat with the tiny botnet he leased with his allowance money.



Extortion via DDoS on the rise

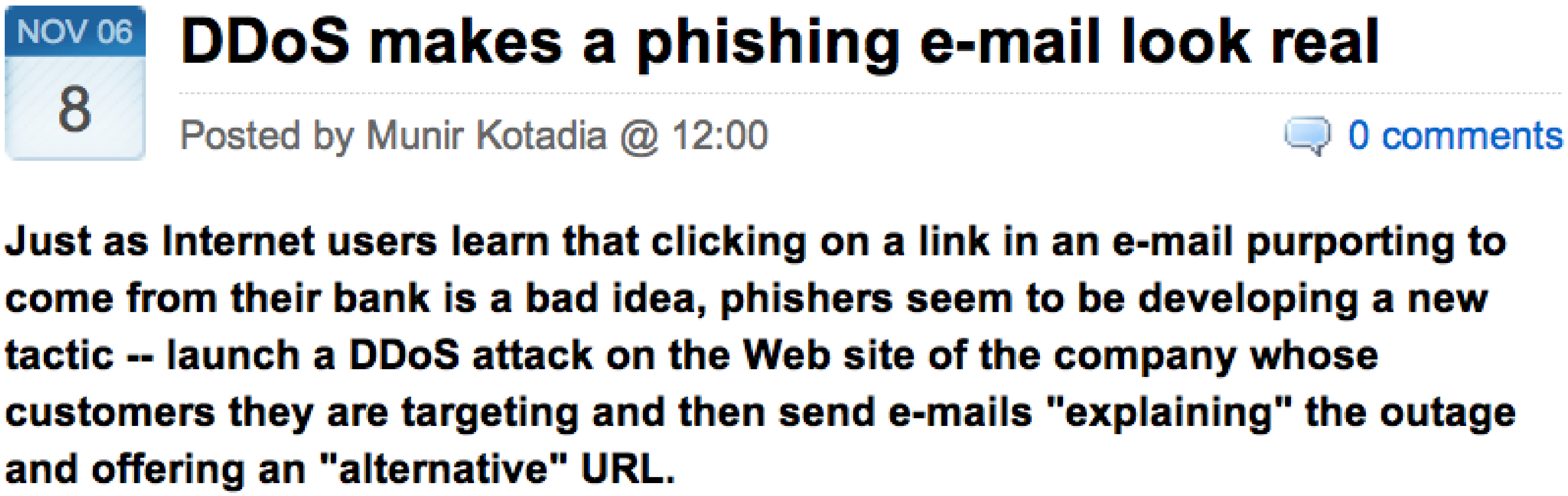
By Denise Pappalardo and Ellen Messmer, Network World, 05/16/05

Criminals are increasingly targeting corporations with distributed denial-of-service attacks designed not to disrupt business networks but to extort thousands of dollars from the companies.



Ivan Maksakov, Alexander Petrov and Denis Stepanov were accused of receiving $4 million from firms that they threatened with cyberattacks.

The trio concentrated on U.K. Internet gambling sites, according to the prosecution. One bookmaker, which refused to pay a demand for $10,000, was attacked and brought offline--which reportedly cost it more than $200,000 a day in lost business.



NOV 06

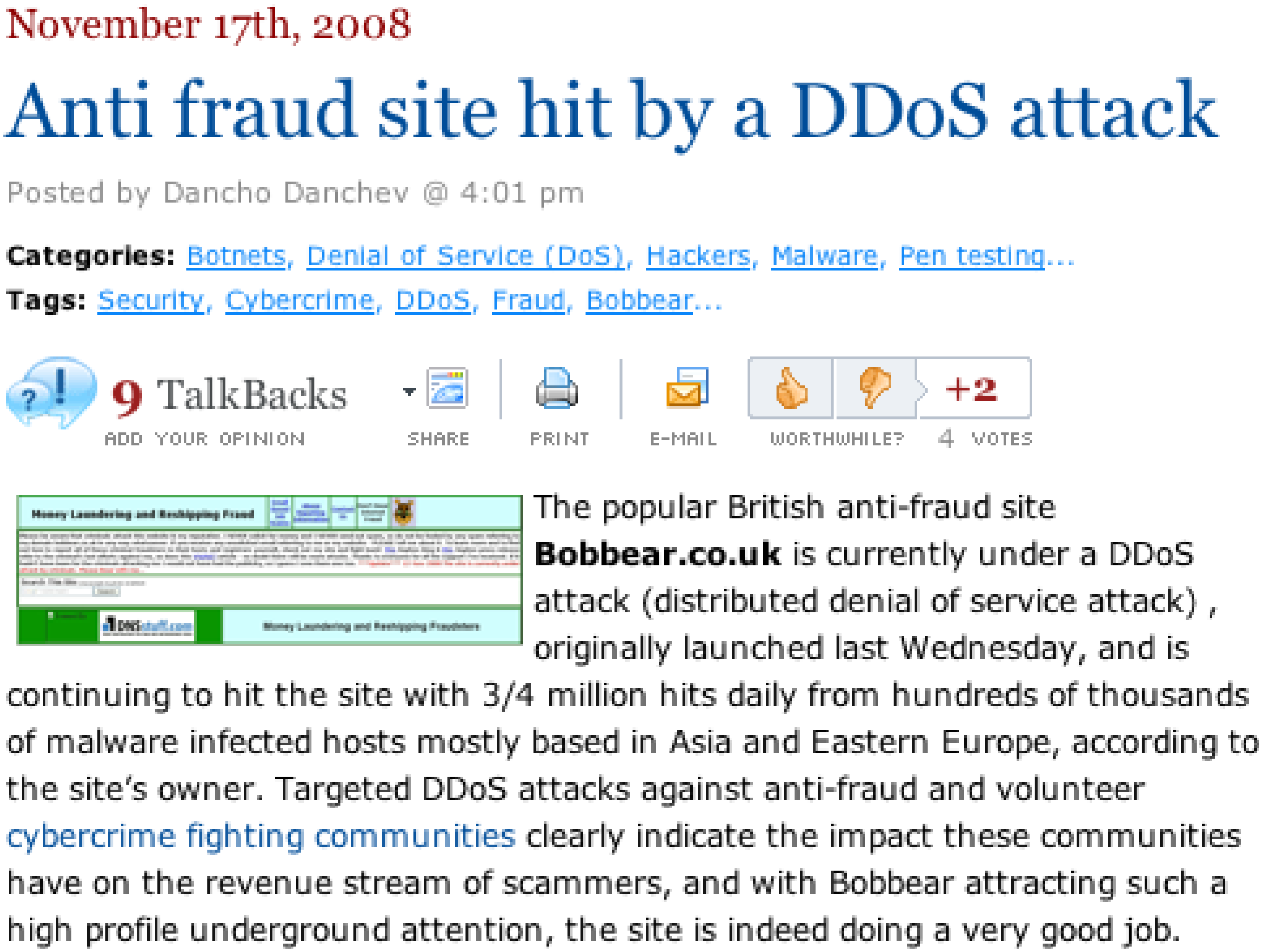
8

DDoS makes a phishing e-mail look real

Posted by Munir Kotadia @ 12:00

0 comments

Just as Internet users learn that clicking on a link in an e-mail purporting to come from their bank is a bad idea, phishers seem to be developing a new tactic -- launch a DDoS attack on the Web site of the company whose customers they are targeting and then send e-mails "explaining" the outage and offering an "alternative" URL.



November 17th, 2008

Anti fraud site hit by a DDoS attack

Posted by Dancho Danchev @ 4:01 pm

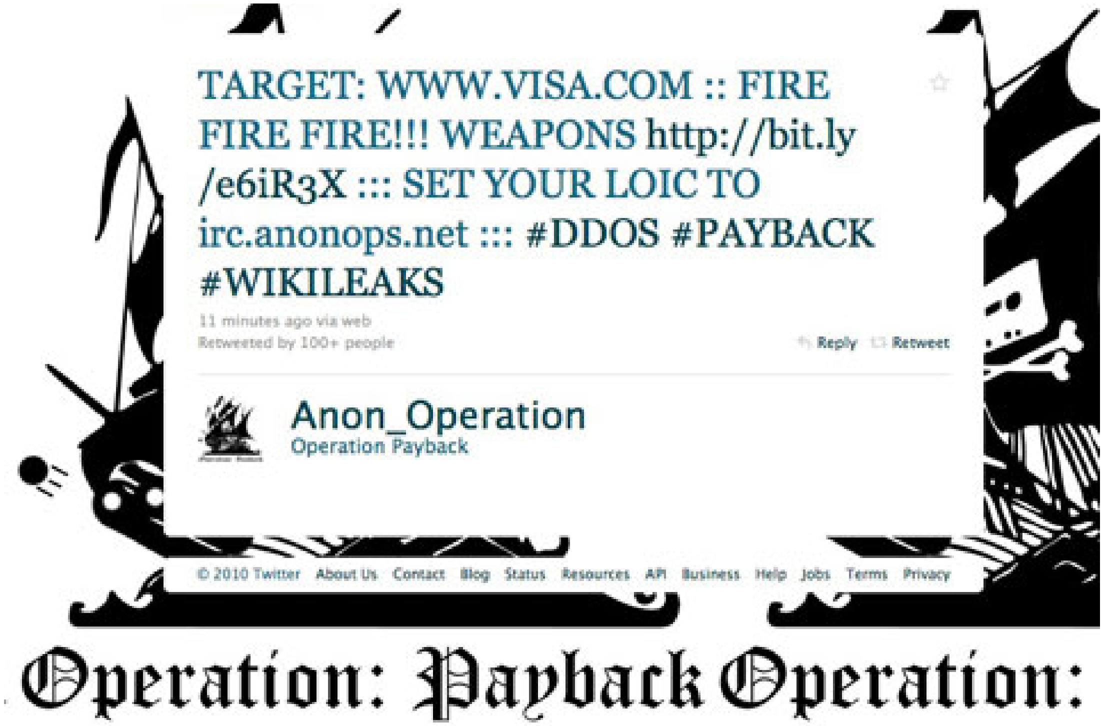
Categories: Botnets, Denial of Service (DoS), Hackers, Malware, Pen testing...
Tags: Security, Cybercrime, DDoS, Fraud, Bobbear...

The popular British anti-fraud site **Bobbear.co.uk** is currently under a DDoS attack (distributed denial of service attack) , originally launched last Wednesday, and is continuing to hit the site with 3/4 million hits daily from hundreds of thousands of malware infected hosts mostly based in Asia and Eastern Europe, according to the site's owner. Targeted DDoS attacks against anti-fraud and volunteer cybercrime fighting communities clearly indicate the impact these communities have on the revenue stream of scammers, and with Bobbear attracting such a high profile underground attention, the site is indeed doing a very good job.

December 8, 2010, 4:18 PM

‘Operation Payback’ Attacks Fell Visa.com

By ROBERT MACKEY



TARGET: WWW.VISA.COM :: FIRE FIRE FIRE!!! WEAPONS http://bit.ly/e6iR3X ::: SET YOUR LOIC TO irc.anonops.net ::: #DDOS #PAYBACK #WIKILEAKS

11 minutes ago via web
Retweeted by 100+ people

Anon_Operation
Operation Payback

Reply
Retweet
© 2010 Twitter About Us Contact Blog Status Resources API Business Help Jobs Terms Privacy
Operation: Payback Operation:

A message posted on Twitter by a group of Internet activists announcing the start of an attack on Visa’s Web site, in retaliation for the company’s actions against WikiLeaks.

Last Updated | 6:54 p.m. A group of Internet activists took credit for crashing the Visa.com Web site on Wednesday afternoon, hours after they launched a similar attack on MasterCard. The cyber attacks, by activists who call themselves Anonymous, are aimed at punishing companies that have acted to stop the flow of donations to WikiLeaks in recent days.

The group explained that its distributed denial of service attacks — in which they essentially flood Web sites site with traffic to slow them down or knock them offline — were part of a broader effort called Operation Payback, which

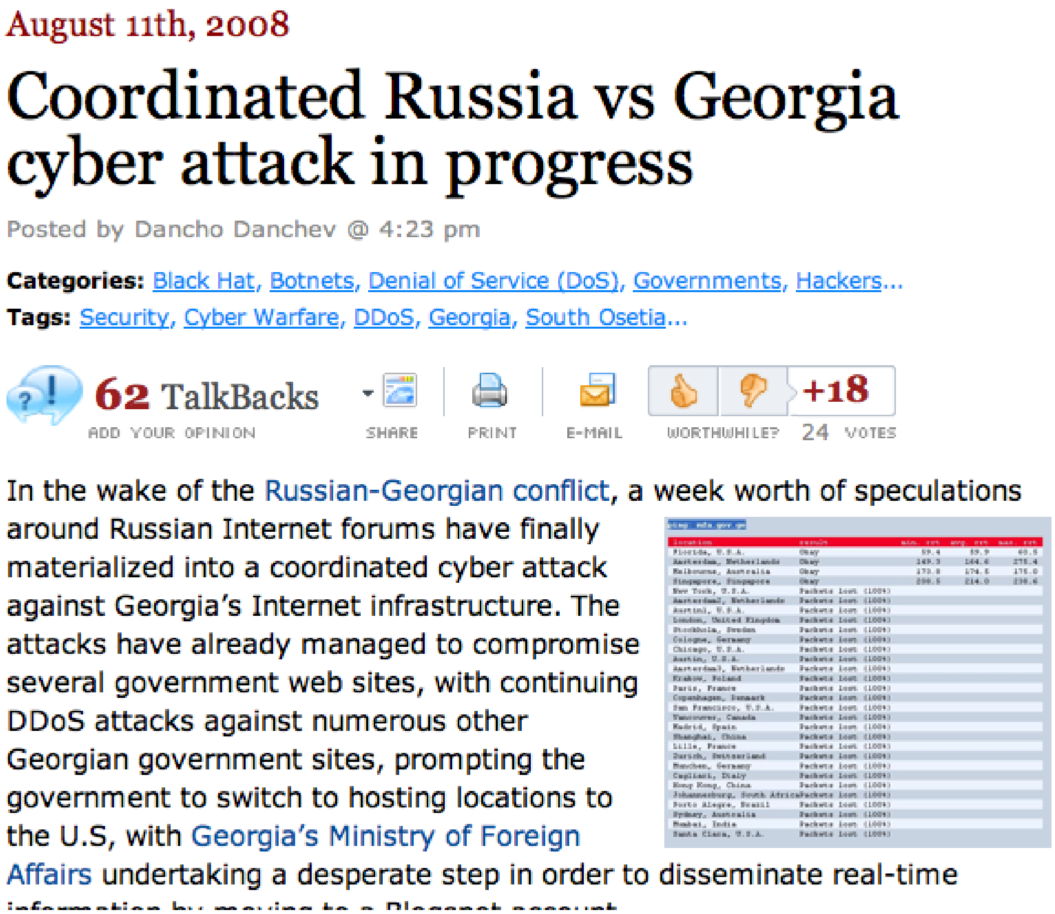
Russia accused of unleashing cyberwar to disable Estonia

- Parliament, ministries, banks, media targeted
- Nato experts sent in to strengthen defences

Ian Traynor in Brussels
The Guardian, Thursday 17 May 2007
Article history

Bronze Soldier, the Soviet war memorial removed from Tallinn. Nisametdinov/AP

A three-week wave of massive cyber-attacks on the small Baltic country of Estonia, the first known incidence of such an assault on a state, is causing alarm across the western alliance, with Nato urgently examining the offensive and its implications.



August 11th, 2008

Coordinated Russia vs Georgia cyber attack in progress

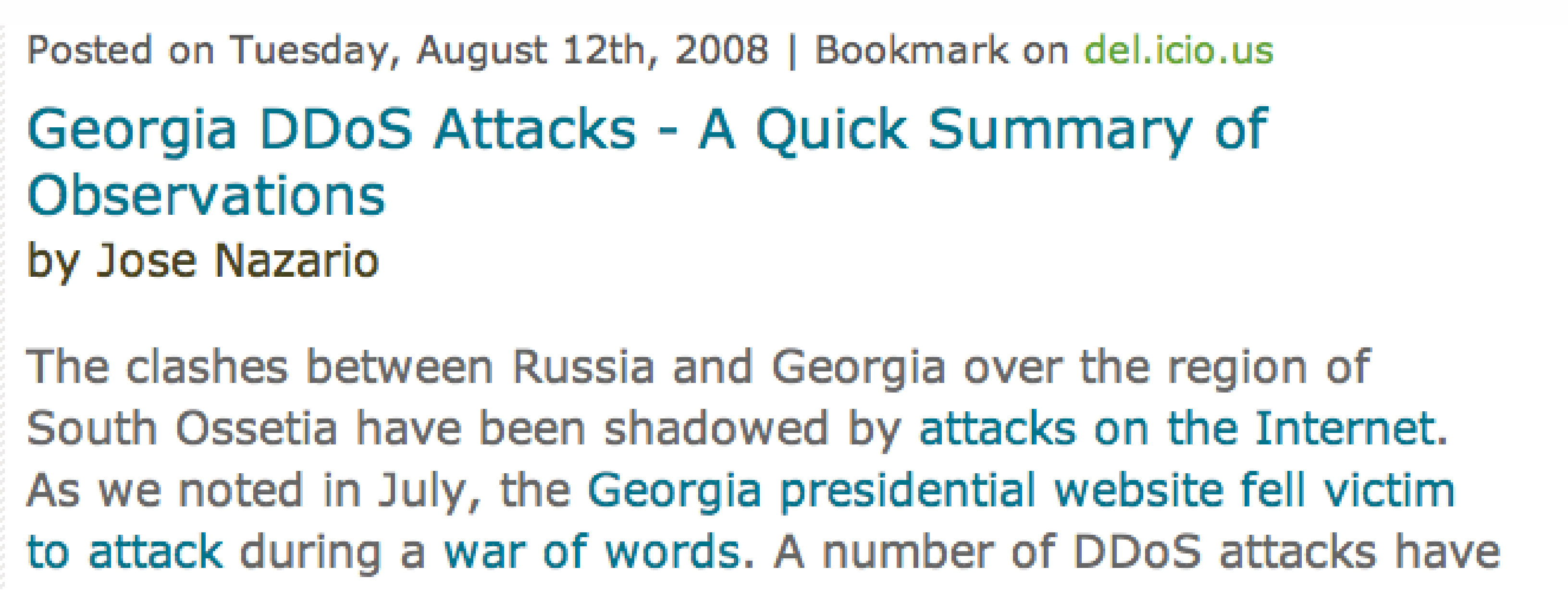
Posted by Dancho Danchev @ 4:23 pm

Categories: Black Hat, Botnets, Denial of Service (DoS), Governments, Hackers...
Tags: Security, Cyber Warfare, DDoS, Georgia, South Osetia...

In the wake of the Russian-Georgian conflict, a week worth of speculations around Russian Internet forums have finally materialized into a coordinated cyber attack against Georgia's Internet infrastructure. The attacks have already managed to compromise several government web sites, with continuing DDoS attacks against numerous other Georgian government sites, prompting the government to switch to hosting locations to the U.S, with Georgia's Ministry of Foreign Affairs undertaking a desperate step in order to disseminate real-time information by moving to a Blogspot account.

ping: mfa.gov.ge

location	result	min. rrt	avg. rrt	max. rrt
Florida, U.S.A.	Okay	59.4	59.9	60.5
Amsterdam, Netherlands	Okay	149.3	164.6	275.4
Melbourne, Australia	Okay	173.8	174.5	175.0
Singapore, Singapore	Okay	206.5	214.0	238.6
New York, U.S.A.	Packets lost (100%)			
Amsterdam2, Netherlands	Packets lost (100%)			
Austin1, U.S.A.	Packets lost (100%)			
London, United Kingdom	Packets lost (100%)			
Stockholm, Sweden	Packets lost (100%)			
Cologne, Germany	Packets lost (100%)			
Chicago, U.S.A.	Packets lost (100%)			
Austin, U.S.A.	Packets lost (100%)			
Amsterdam3, Netherlands	Packets lost (100%)			
Krakow, Poland	Packets lost (100%)			
Paris, France	Packets lost (100%)			
Copenhagen, Denmark	Packets lost (100%)			
San Francisco, U.S.A.	Packets lost (100%)			
Vancouver, Canada	Packets lost (100%)			
Madrid, Spain	Packets lost (100%)			
Shanghai, China	Packets lost (100%)			
Lille, France	Packets lost (100%)			
Zurich, Switzerland	Packets lost (100%)			
Munchen, Germany	Packets lost (100%)			
Cagliari, Italy	Packets lost (100%)			
Hong Kong, China	Packets lost (100%)			
Johannesburg, South Africa	Packets lost (100%)			
Porto Alegre, Brazil	Packets lost (100%)			
Sydney, Australia	Packets lost (100%)			
Mumbai, India	Packets lost (100%)			
Santa Clara, U.S.A.	Packets lost (100%)			

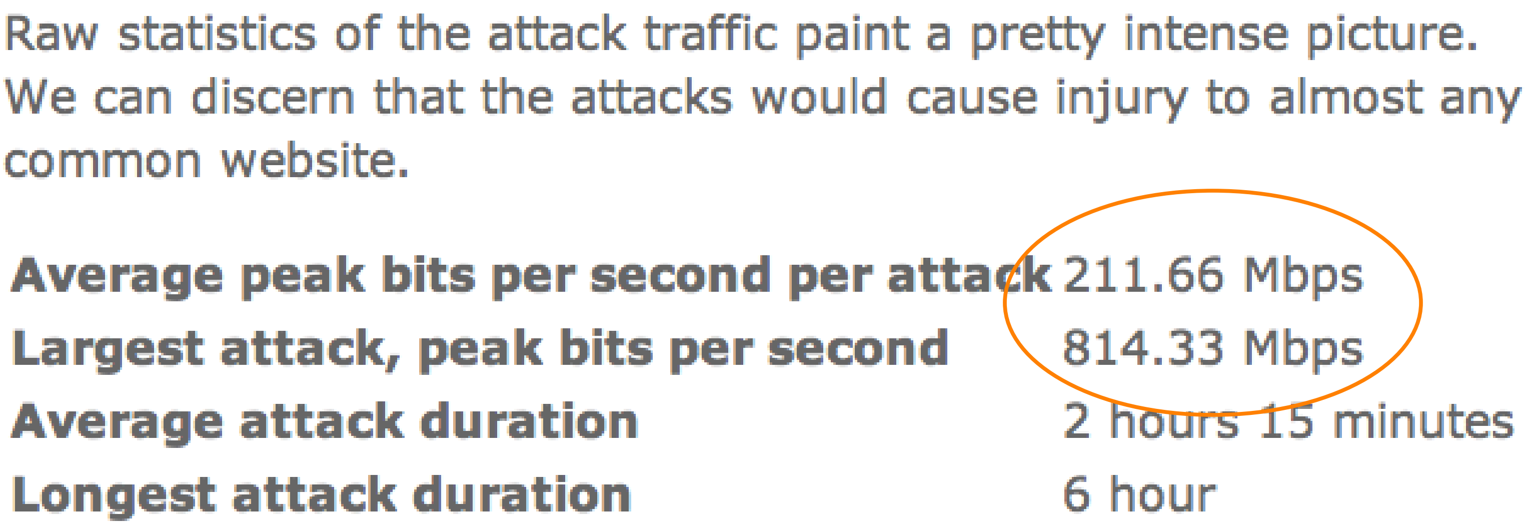


Posted on Tuesday, August 12th, 2008 | Bookmark on del.icio.us

Georgia DDoS Attacks - A Quick Summary of Observations

by Jose Nazario

The clashes between Russia and Georgia over the region of South Ossetia have been shadowed by attacks on the Internet. As we noted in July, the Georgia presidential website fell victim to attack during a war of words. A number of DDoS attacks have



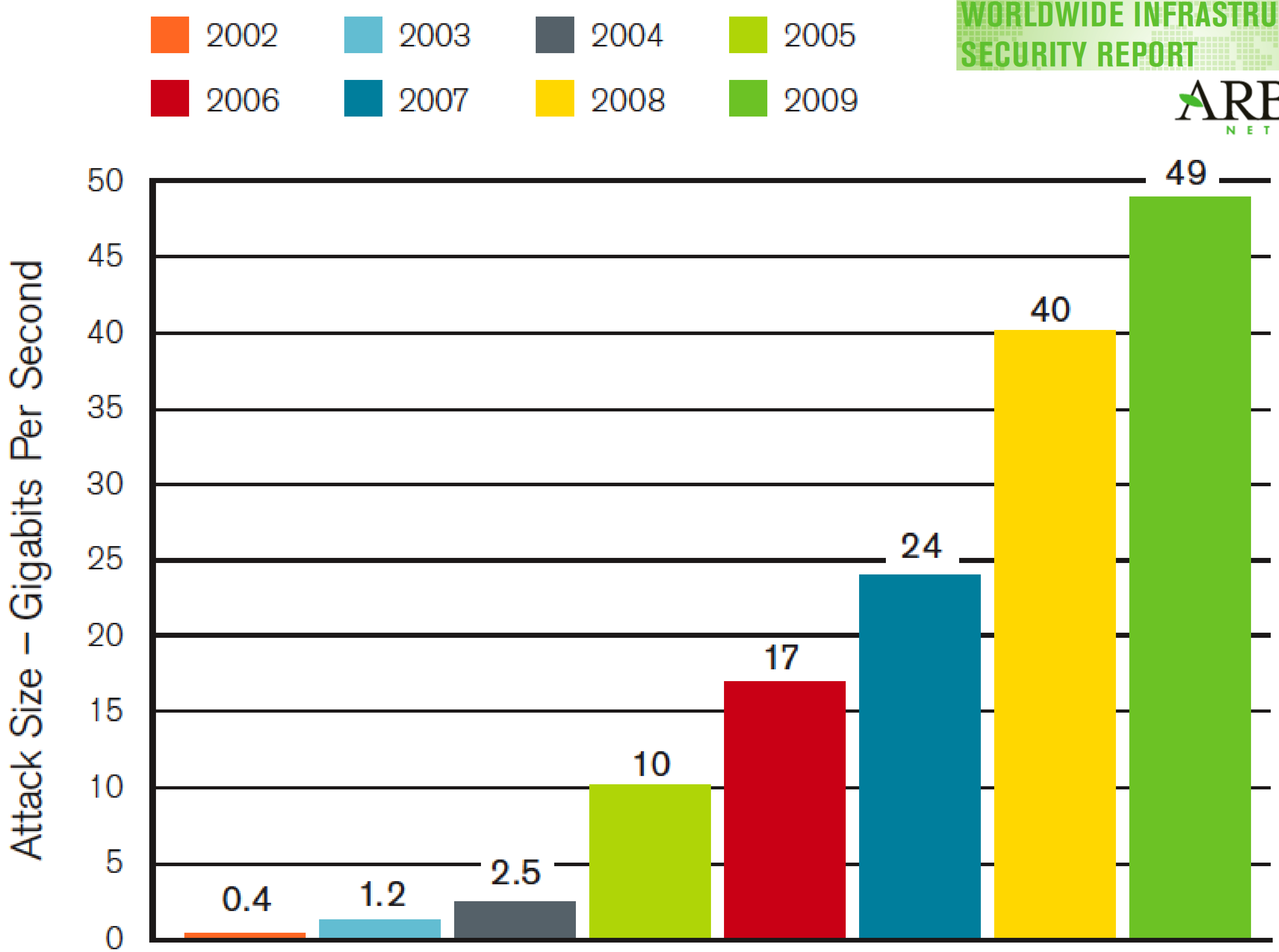
Raw statistics of the attack traffic paint a pretty intense picture. We can discern that the attacks would cause injury to almost any common website.

Average peak bits per second per attack	211.66 Mbps
Largest attack, peak bits per second	814.33 Mbps
Average attack duration	2 hours 15 minutes
Longest attack duration	6 hour

Largest DDoS Attack – 49 Gigabits Per Second

WORLDWIDE INFRASTRUCTURE SECURITY REPORT

ARBOR NETWORKS



It’s Not A “Level Playing Field”

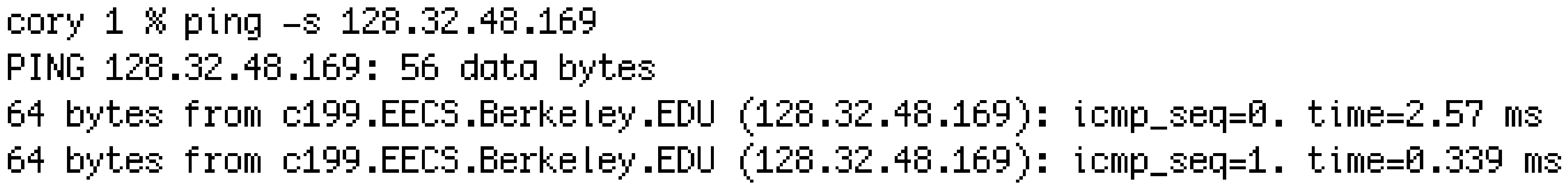
- When defending resources from exhaustion, need to beware of asymmetries, where attackers can consume victim resources with little comparable effort
  - Makes DoS easier to launch
  - Defense costs much more than attack

- Particularly dangerous form of asymmetry: **amplification**
  - Attacker leverages system’s own structure to pump up the load they induce on a resource

Amplification: Network DoS

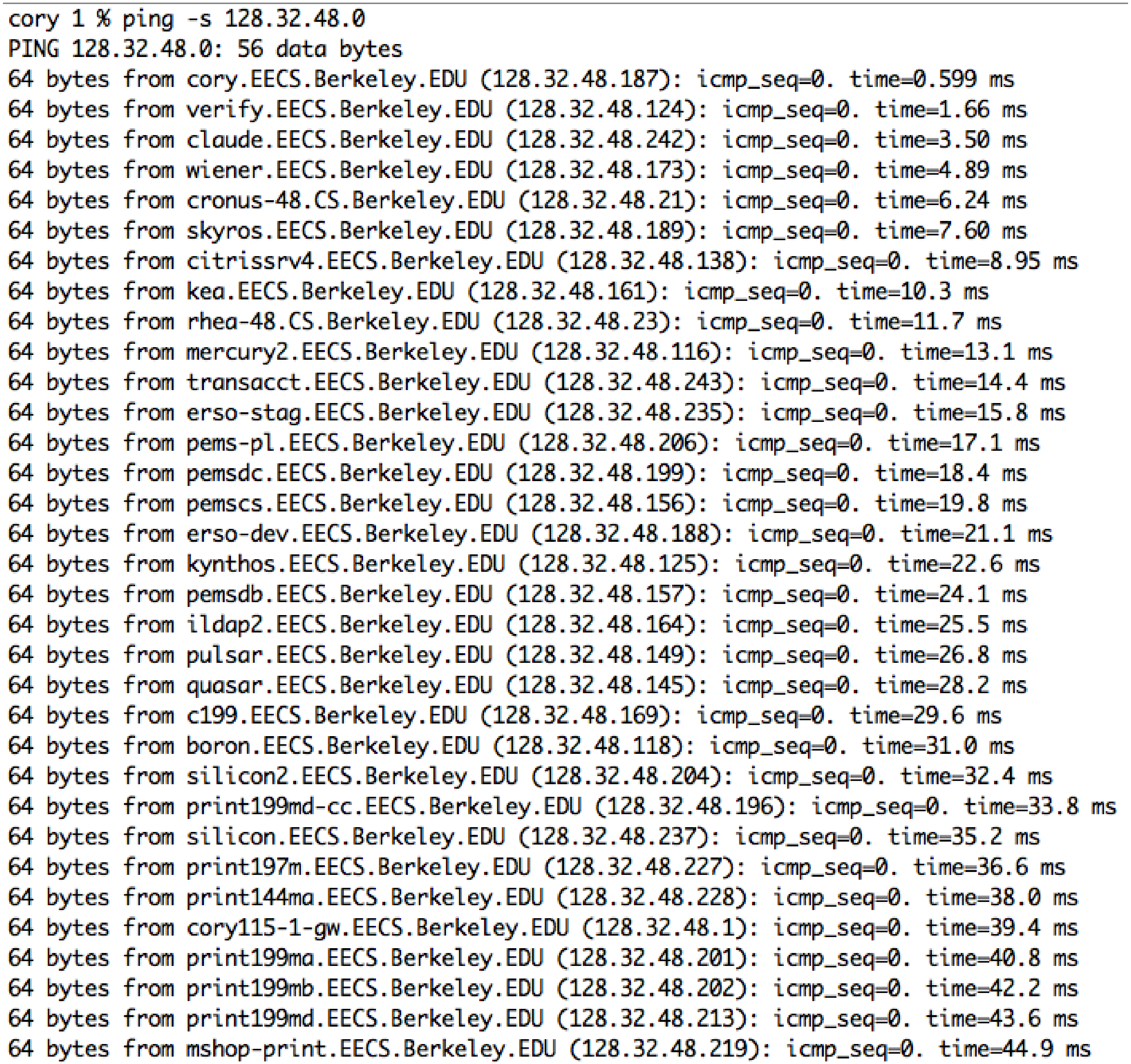
- One technique for magnifying flood traffic: leverage Internet's *broadcast functionality*

```
cory 1 % ping -s 128.32.48.169
PING 128.32.48.169: 56 data bytes
```



```
cory 1 % ping -s 128.32.48.169
PING 128.32.48.169: 56 data bytes
64 bytes from c199.EECS.Berkeley.EDU (128.32.48.169): icmp_seq=0. time=2.57 ms
64 bytes from c199.EECS.Berkeley.EDU (128.32.48.169): icmp_seq=1. time=0.339 ms
```

```
cory 1 % ping -s 128.32.48.0
PING 128.32.48.0: 56 data bytes
```



```
cory 1 % ping -s 128.32.48.0
PING 128.32.48.0: 56 data bytes
64 bytes from cory.EECS.Berkeley.EDU (128.32.48.187): icmp_seq=0. time=0.599 ms
64 bytes from verify.EECS.Berkeley.EDU (128.32.48.124): icmp_seq=0. time=1.66 ms
64 bytes from claude.EECS.Berkeley.EDU (128.32.48.242): icmp_seq=0. time=3.50 ms
64 bytes from wiener.EECS.Berkeley.EDU (128.32.48.173): icmp_seq=0. time=4.89 ms
64 bytes from cronus-48.CS.Berkeley.EDU (128.32.48.21): icmp_seq=0. time=6.24 ms
64 bytes from skyros.EECS.Berkeley.EDU (128.32.48.189): icmp_seq=0. time=7.60 ms
64 bytes from citrissrv4.EECS.Berkeley.EDU (128.32.48.138): icmp_seq=0. time=8.95 ms
64 bytes from kea.EECS.Berkeley.EDU (128.32.48.161): icmp_seq=0. time=10.3 ms
64 bytes from rhea-48.CS.Berkeley.EDU (128.32.48.23): icmp_seq=0. time=11.7 ms
64 bytes from mercury2.EECS.Berkeley.EDU (128.32.48.116): icmp_seq=0. time=13.1 ms
64 bytes from transacct.EECS.Berkeley.EDU (128.32.48.243): icmp_seq=0. time=14.4 ms
64 bytes from erso-stag.EECS.Berkeley.EDU (128.32.48.235): icmp_seq=0. time=15.8 ms
64 bytes from pems-pl.EECS.Berkeley.EDU (128.32.48.206): icmp_seq=0. time=17.1 ms
64 bytes from pemsdc.EECS.Berkeley.EDU (128.32.48.199): icmp_seq=0. time=18.4 ms
64 bytes from pemscs.EECS.Berkeley.EDU (128.32.48.156): icmp_seq=0. time=19.8 ms
64 bytes from erso-dev.EECS.Berkeley.EDU (128.32.48.188): icmp_seq=0. time=21.1 ms
64 bytes from kynthos.EECS.Berkeley.EDU (128.32.48.125): icmp_seq=0. time=22.6 ms
64 bytes from pemsdb.EECS.Berkeley.EDU (128.32.48.157): icmp_seq=0. time=24.1 ms
64 bytes from ildap2.EECS.Berkeley.EDU (128.32.48.164): icmp_seq=0. time=25.5 ms
64 bytes from pulsar.EECS.Berkeley.EDU (128.32.48.149): icmp_seq=0. time=26.8 ms
64 bytes from quasar.EECS.Berkeley.EDU (128.32.48.145): icmp_seq=0. time=28.2 ms
64 bytes from c199.EECS.Berkeley.EDU (128.32.48.169): icmp_seq=0. time=29.6 ms
64 bytes from boron.EECS.Berkeley.EDU (128.32.48.118): icmp_seq=0. time=31.0 ms
64 bytes from silicon2.EECS.Berkeley.EDU (128.32.48.204): icmp_seq=0. time=32.4 ms
64 bytes from print199md-cc.EECS.Berkeley.EDU (128.32.48.196): icmp_seq=0. time=33.8 ms
64 bytes from silicon.EECS.Berkeley.EDU (128.32.48.237): icmp_seq=0. time=35.2 ms
64 bytes from print197m.EECS.Berkeley.EDU (128.32.48.227): icmp_seq=0. time=36.6 ms
64 bytes from print144ma.EECS.Berkeley.EDU (128.32.48.228): icmp_seq=0. time=38.0 ms
64 bytes from cory115-1-gw.EECS.Berkeley.EDU (128.32.48.1): icmp_seq=0. time=39.4 ms
64 bytes from print199ma.EECS.Berkeley.EDU (128.32.48.201): icmp_seq=0. time=40.8 ms
64 bytes from print199mb.EECS.Berkeley.EDU (128.32.48.202): icmp_seq=0. time=42.2 ms
64 bytes from print199md.EECS.Berkeley.EDU (128.32.48.213): icmp_seq=0. time=43.6 ms
64 bytes from mshop-print.EECS.Berkeley.EDU (128.32.48.219): icmp_seq=0. time=44.9 ms
```

Amplification: Network DoS

- One technique for magnifying flood traffic: leverage Internet's *broadcast functionality*
- How does an attacker exploit this?
  - Send traffic to the broadcast address and spoof it *as though the DoS victim sent it*
  - All of the replies then go to the victim rather than the attacker's machine
  - Each attacker pkt yields dozens of flooding pkts
- Another example: DNS lookups
  - *Reply is often much bigger than request*
  - So attacker spoofs request seemingly from the target
    - Small attacker packet yields large flooding packet

smurf attack