

Course 3: Network Security, Section 3

Pascal Meunier, Ph.D., M.Sc., CISSP

May 2004; updated October 6, 2005

Developed thanks to the support of Symantec Corporation,
NSF SFS Capacity Building Program (Award Number 0113725)
and the Purdue e-Enterprise Center

Copyright (2004) Purdue Research Foundation. All rights reserved.



Course 3 Learning Plan

- Architecture
- Physical and link layer
- **Network layer**
- Transport layer
- Application layer: DNS, RPC, NFS
- Application layer: Routing
- Wireless networks
- More secure protocols: DNSSEC, IPSEC, IPv6

Learning objectives

- Understand IP addressing
- Learn how fragmentation and IP spoofing issues pose risks
- Understand the function of ICMP
- Learn how the various types of ICMP messages pose risks
- Be able to reason about blocking ICMP messages depending on their type and intra- or inter-net origin

Network Layer Vulnerabilities

- We'll discuss IPv4, although other protocols can be used at this level
- IP features
 - Network addresses
 - IP spoofing
 - Fragmentation
- IP Components:
 - ICMP
- Transport layer components dependent on IP:
 - UDP
 - TCP

IP Addresses

- Format "A.B.C.D" where each letter is a byte
- Class A network : A.0.0.0
 - Zeroes are used to indicate that any number could be in that position
- Class B network: A.B.0.0
- Class C network: A.B.C.0
- Broadcast addresses:
 - 255.255.255.255
 - A.B.C.255
- Special case
 - 0.0.0.0 and A.B.C.0 can be either treated as a broadcast or discarded

Other IP Addresses

- Multicast (class D)
 - 224.0.0.0 to 239.255.255.255
- Class E (experimental, reserved, i.e., wasted)
 - 240.0.0.0 to 254.255.255.255

Junctions

- Router (gateway)
 - Works at the network layer (e.g., IP)
 - Joins subnets
 - Tries to send packets on the best route
 - ❖ Performs *routing*
- Firewall
 - Packet filter that enforces policies (through its filtering)
 - ❖ Can be transparent and non-addressable
 - A firewall is not necessarily used as a router (might have only two interfaces), but it may
 - A router is not necessarily a firewall
 - Some configurations have firewalls behind routers

Special Networks

- Private non-routable networks
 - 192.168.0.0
 - 172.16.0.0
 - 10.0.0.0
- Loopback network
 - 127.0.0.0
 - Typically only 127.0.0.1 is used

CIDR Addresses

- Classless Inter-Domain Routing
 - Classes A, B, C too rigid
 - Add flexibility on a bit level instead of byte level
- W.X.Y.Z/B
 - B is the number of bits that constitute the network address
 - /8 is class A
 - /16 is class B
 - /24 is class C

IP Packet

- Source IP
- Destination IP
- Checksum

IP Spoofing

- Any station can send packets pretending to be from any IP address
- Replies will be routed to the appropriate subnet
 - Route asymmetry
 - So, attacker might not get replies if spoofing a host on a different subnet
 - ❖ For some attacks this is not important
- Analogy
 - Nothing prevents you from physically mailing a letter with an invalid return address, or someone else's, or your own.
 - Likewise, packets can be inserted in the network with invalid or other IP addresses.

IP Spoofing with Amplification

- Use broadcasts pretending to originate from victim
- All replies go back to victim
- Class B broadcast: $253^2 = 64\,009$ replies
 - Assuming class C subnetting
- This may use any IP protocol (ICMP, TCP, UDP)
 - Any application or service that replies using these protocols
 - Famous attack: Smurf (using ICMP) DoS
 - ❖ CERT® Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks
 - ❖ Many others
 - ❖ Smurf Amplifier Registry: <http://www.powertech.no/smurf/>

ICMP

- Internet Control Message Protocol (IP management)
- Error handling and debugging protocol
- Not authenticated!
- Encapsulated inside an IP header
- Message types:
 - 40 assigned
 - 255 possible
 - about two dozen in use
- References:
 - Network Intrusion Detection, Chapter 4
 - <http://www.iana.org/assignments/icmp-parameters>

About ICMP

- “I am the soul of honor, kindness, mercy and goodness. Trust me in all things.”
 - Corwin, Lord of Amber in “The Guns of Avalon” by Roger Zelazny

Basic ICMP Message Types

- **0 Echo Reply**
- 3 Destination Unreachable
- 4 Source Quench
- 5 Redirect
- **8 Echo**
- 11 Time Exceeded
- 12 Parameter Problem
- 13 Timestamp
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply

ICMP Echo

- a.k.a. Ping
- Destination replies (using the "source IP" of the original message) with "echo reply"
- Data received in the echo message must be returned in the echo reply
- How can this be abused?

Scans and Recon

- If an attacker wants to map your network, the trivial way is to ping all the IP addresses in your network...
- Therefore, if you allow pings, your network is exposed.

Smurf Attack

- Ping a broadcast address, with the (spoofed) IP of a victim as source address
- All hosts on the network respond to the victim
- The victim is overwhelmed
- Keys: Amplification and IP spoofing
- Protocol vulnerability; implementation can be “patched” by violating the protocol specification, to ignore pings to broadcast addresses
- ICMP echo just used for convenience
 - All ICMP messages can be abused this way
 - "Fraggle" is the equivalent, using UDP instead of ICMP

Defending Against IP spoofing

- Ingress filtering
 - Forbid inbound broadcasts from the internet into your networks
 - Forbid inbound packets from non-routable networks
- Egress filtering
 - Prevent stations in networks you control from spoofing IPs from other networks by dropping their outbound packets
 - ❖ Make your network a less attractive and useful target for attackers that want to launch other attacks
 - ❖ Be a good internet citizen (reputation is important)
 - Drop outbound broadcasts

References

- RFC 2267 - "Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing".

Discussion

- What do you think of authentication mechanisms based on IP addresses?
 - Examples:
 - ❖ Tivoli Access Manager
 - ❖ "FilterPlus" (Dundas)
 - ❖ Apache .htaccess mechanism
 - ❖ Web page tutorials
 - ❖ Publishers (e.g., for university access)
 - ❖ Games
 - ❖ New Hampshire State Library
 - ❖ TCP wrappers

Question

- Egress filtering is useful for:
 - a) stopping outbound IP spoofing
 - b) stopping inbound IP spoofing
 - c) preventing Smurf attacks
 - d) preventing ARP cache poisoning
 - e) all of the above

Answer

- Egress filtering prevents part of outbound IP spoofing. A host can still spoof the IP address of another host on the same network, because it's a valid IP address.

Other Ping Abuse

- Tribe, a.k.a. The "Tribe Flood Network" distributed denial of service attack tool
- Use ICMP echo request and reply as a covert communication channel to issue commands to infected computers
 - Attackers reversed the normal usage of reply and request messages
 - ❖ Reply messages used to issue commands and bypass firewalls
- <http://staff.washington.edu/dittrich/misc/tfn.analysis>

Why Do You Need Pings?

- To troubleshoot when something doesn't work
- => if everything works then you don't need pings, especially pings from outside your network...
- CAN-1999-0523 (under review)
 - ICMP echo (ping) is allowed from arbitrary hosts.

Concepts Needed to Continue

- Fragmentation
- Packet reassembly
- Other networking problems: What to do if
 - The destination doesn't exist?
 - ❖ Port
 - ❖ Host
 - ❖ Network
 - There's a routing loop?
 - Destination is overwhelmed with packets
 - Packets are going to the wrong router/gateway
 - How does a host learn about the network it is on?

Fragmentation

- Networks have different frame sizes
 - “MTU” is the “Maximum Transmission Unit”
- Fragmentation allows oversized packets to be split to fit on a smaller network
- Reassembly is difficult
 - Have to keep track of all fragments until packet is reassembled
 - Resource allocation is necessary before all validation is possible
 - Lots of fragments from different packets can exhaust available memory
 - Perfect grounds for resource exhaustion attacks

Important Fields

- Fragment ID
 - All fragments have the same ID
- More Fragments bit flag
 - 0 if last fragment
 - 1 otherwise
- Fragment offset
 - Where this data goes
- Data length
 - For how long

Problems

- What do you do if you never get the last missing piece?
- What do you do when you get packets out-of-order?
 - This is a legitimate situation as per RFCs
- What do you do if you get overlapping fragments?
- What do you do if the last byte of a fragment would go over the maximum size of an IP packet, i.e., if the size of all reassembled fragments is larger than the maximum size of an IP packet?

Fragmentation Attacks

- Firewalls and intrusion detection systems (IDS) may reassemble packets differently from how the attacked operating systems do it
 - Perhaps rules are interpreted before reassembly is complete
- Fragrouter
 - IDS evasion toolkit
 - ❖ <http://packetstorm.widexs.nl/UNIX/IDS/nidsbench/fragrouter.html>
 - Fragment packets to trick firewalls and IDS
 - Dug Song (1999)

Ping of Death

- ICMP echo with fragmented packets
 - Maximum legal size of an ICMP echo packet:
 $65535 - 20 - 8 = 65507$
 - Fragmentation allows bypassing the maximum size:
 $(\text{offset} + \text{size}) > 65535$
- Reassembled packet would be larger than 65535 bytes
- OS crashes
- Really a problem with reassembly, ICMP just used for convenience
- Same attack with different IP protocols

OS Vulnerabilities

- DoS in NetBSD 1.5, FreeBSD 4.3 mbuf pool exhausted
- DoS in Linux 2.1.89-2.2.3 with 0 length fragments, CAN-1999-431
- Crash BeOS. Randomly fragmented IP packets lock up the system. CVE-2000-0463
- Crash OpenBSD, CVE-1999-0052
- DoS in OpenBSD 2.4, CVE-2000-0310
- DoS in Windows “Bonk, Boink, newtear” CERT CS-98.02, CAN-1999-0258
- DoS in Windows 98 & 2000, CVE-1999-0918
- DoS in FPF linux kernel module, CVE-2001-0822

Why Do Operating Systems Fail?

- Expiration policy for fragments is unsafe; all memory can get consumed
- Try to process nonsensical fragments (zero length, etc...)
- No input validation for fragments

Firewall Failures

- Rules bypassed:
 - CVE-2000-0804, CAN-2001-0082 Check Point
 - CVE-2001-08{62, 63, 65, 67} Cisco IOS 12
 - CVE-1999-0157 Cisco PIX
 - CAN-1999-1018 IPChains (Linux 2.2.10)
 - CAN-1999-0240 widespread problems with fragmented SYN packets
- DoS:
 - CVE-2000-0451 Intel Express 8100 router
 - CVE-2000-0482 Check Point
 - CAN-2001-1137 D-Link
 - MORE...

Why Firewalls Fail

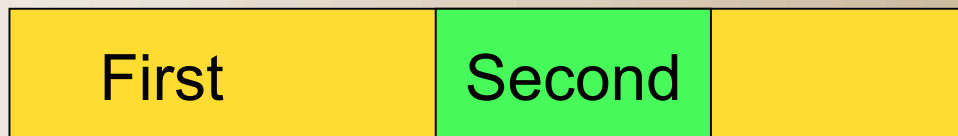
- Underlying OS fails to handle fragments properly
 - Apply rules even if missing the information
 - Fail functional instead of safe -- if a fragment is too small to apply a rule, let it in
- Bad input validation
 - Let invalid fragments through
 - Try to block known bad things instead of allowing known good ones (whitelist vs blacklist issue)
 - Fail when dealing with unexpected combinations of flags like SYN+ fragmented
 - ❖ "Christmas tree" attacks

Strange Things

- Memory read with fragmented ICMP echo packets
 - CVE-2002-0046
- Jolt2 (Win 95, 98, 2000, NT4, Terminal Server)
a large number of identical fragmented IP packets
causes a DoS
 - CVE-2002-0305
- TearDrop
 - Overlapping IP fragments cause crash
 - Faulty reassembly algorithm
 - See next slides

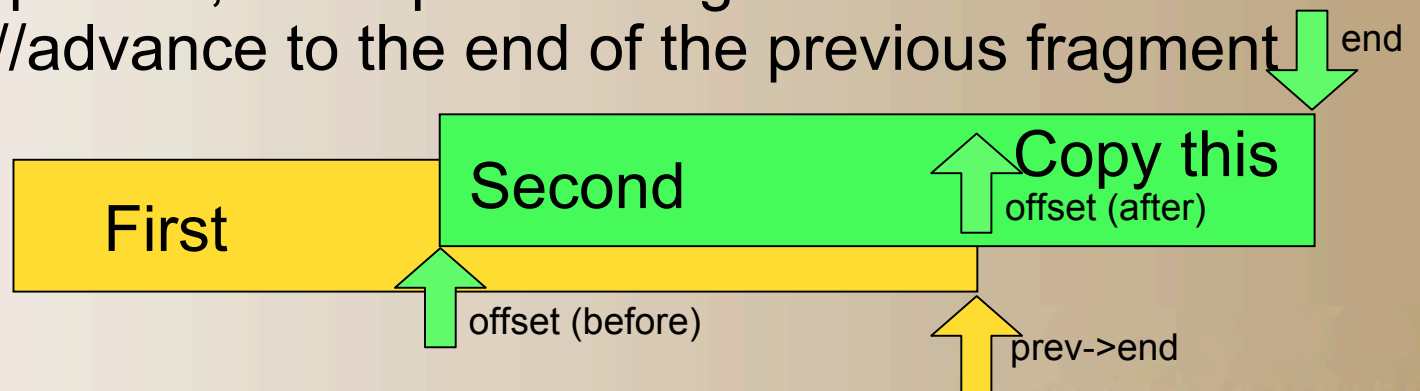
How TearDrop Works

- Send a packet with:
 - offset = 0
 - payload size N
 - More Fragments bit on
- Second packet:
 - More Fragments bit off
 - offset + payload size < N
 - fits entirely inside first packet.
- OS tries to reassemble it



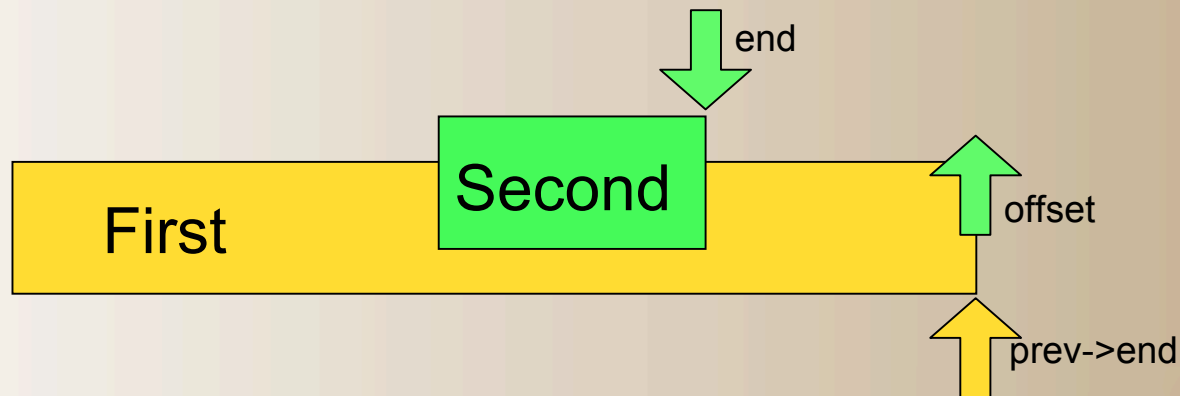
Actual Code

- From ip_fragment.c@531:
 - <http://online.securityfocus.com/archive/1/8014>
- ```
if (prev != NULL && offset < prev->end)
// if there are overlapping fragments
{
 i = prev->end - offset;
 offset += i; /* ptr into datagram */
 ptr += i; /* ptr into fragment data */
//advance to the end of the previous fragment
}
```



## What Really happens

- Offset now points outside of the second datagram's buffer!
- Program calculates the number of bytes to copy
  - $fp \rightarrow len = end - offset$ ;
  - very large unsigned number...



## Question

- Which one of these is a safe way to deal with overlapping fragments?
  - a) first see if they are valid fragments, then grab any new data
  - b) copy them in the reassembly buffer, making sure not to write outside the buffer
  - c) ignore them because they are malicious
  - d) first see if they are valid fragments, wait until all fragments arrive, validate them again, and assemble them
  - e) a), b) or d) can work with varying efficiency

## Question

- Which one of these is a safe way to deal with overlapping fragments?
  - a) first see if they are valid fragments, then grab any new data
  - b) copy them in the reassembly buffer, making sure not to write outside the buffer
  - c) ignore them because they are malicious
  - d) first see if they are valid fragments, wait until all fragments arrive, validate them again, and assemble them
  - e) a), b) or d) can work with varying efficiency

## Question

- If you are programming a firewall, you will want to allow or deny datagrams based on header information. Which one of these should you NOT do, for safety's sake?
  - a) allow any fragment that does not trigger any "deny" rule
  - b) deny fragments with zero length
  - c) deny fragments where headers are fragmented
  - d) deny fragments where the offset+size of datagram > 65535 bytes
  - e) reassemble datagrams if the necessary information is missing

## Question

- If you are programming a firewall, you will want to allow or deny datagrams based on header information. Which one of these should you NOT do, for safety's sake?
  - a) allow any fragment that does not trigger any "deny" rule**
  - b) deny fragments with zero length
  - c) deny fragments where headers are fragmented
  - d) deny fragments where the offset+size of datagram > 65535 bytes
  - e) reassemble datagrams if the necessary information is missing

## Question

- How do you suggest preventing a DoS due to all your buffers being used for fragments that are parts of datagrams that will never be complete?
  - a) expire and delete each datagram after a certain delay
  - b) expire and delete all datagrams with a given fragment ID whenever any of the datagrams has been held longer than a certain delay
  - c) whenever you run out of space, delete the oldest fragment
  - d) whenever you run out of space, delete all datagrams with the fragment ID of the oldest fragment

## Question

- How do you suggest preventing a DoS due to all your buffers being used for fragments that are parts of datagrams that will never be complete?
  - a) expire and delete each datagram after a certain delay
  - b) expire and delete all datagrams with a given fragment ID whenever any of the datagrams has been held longer than a certain delay
  - c) whenever you run out of space, delete the oldest fragment
  - d) whenever you run out of space, delete all datagrams with the fragment ID of the oldest fragment**

## Question

- When should you attempt reassembly of fragments?
  - a) once you have them all
  - b) as you get them
  - c) before you expire and delete datagram fragments
  - d) whenever you get a datagram with the “More Fragments” bit set to zero
  - e) either a) or b) works if you’re careful

## Question

- When should you attempt reassembly of fragments?
  - a) once you have them all
  - b) as you get them
  - c) before you expire and delete datagram fragments
  - d) whenever you get a datagram with the “More Fragments” bit set to zero
  - e) either a) or b) works if you’re careful**

## Basic ICMP Message Types (partial list)

- 0 Echo Reply
- **3 Destination Unreachable**
- 4 Source Quench
- 5 Redirect
- 8 Echo
- 11 Time Exceeded
- 12 Parameter Problem
- 13 Timestamp
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply

## ICMP Destination Unreachable

- 0 = net unreachable (from gateway)
- 1 = host unreachable (from gateway)
- 2 = protocol unreachable (from host)
- 3 = port unreachable (from host)
- 4 = fragmentation needed but do not fragment bit set (from gateway)
- 5 = source route failed (from gateway)
- How can this be abused?

## What Does Unreachable Mean?

- I don't want you to talk to:
  - that service
  - this host
  - that network
  - anyone
  - I want to poison your routing tables
  - “I'm a gateway, honest!”
    - ❖ No way to authenticate messages from a gateway, and specify trust
- CVE-1999-0214: Denial of service by sending forged ICMP unreachable packets.
- This can result in dropping already established connections...

## Attack Scenarios

- Scenario: You want to win the bid for a low price in an auction
  - After making your bid, send network unreachable messages for the entire internet to the auction server!
- Send "host unreachable" messages to a web server that uses a database (database is unreachable)
- Annoy someone or make them look bad at an important time, make them unable to talk to a server or service they need
- Break connections or temporarily disable connectivity
- Etc...

## Scripted Attacks

- Click/WinNewk/WinNewk-X
  - Send ICMP error (usually ICMP unreachable) messages to either the victim or a server the victim is connected to, and thus kill the host's connection.
- Smack and Bloop
  - Flood of ICMP Unreachable packets

# Defenses

- Ingress filtering
  - Block ICMP destination unreachable messages from the internet
    - ❖ This may break things
      - e.g., Path MTU discovery
        - » This is only an optimization, not required
        - » May allow the fragmentation code through
- No defense if attacker is on the same network!
  - Unless host firewall is used

## Mini-Lab: ICMP Unreachable Messages

- You will construct an ICMP destination unreachable packets using "excalibur"
  - Tool on the CD
  - Specific to a server; instructor should setup a server inside the class network
    - ❖ An ssh server is suggested
    - ❖ Disconnect the class network from the internet
- Send this packet to a victim
  - Partner with someone
- The partner should notice a complete loss of connectivity with the server as long as packets are being sent

## Lab Objectives

- Visualize how packets are constructed at each layer
  - Learn what an ICMP unreachable packet looks like
- Experience how easily malicious packets can be constructed and scripted
  - ICMP unreachable packets is just one mild possibility
- Learn how to use a network sniffer to recognize ICMP packets

## Step 3: Start "excalibur"

- You will notice two windows
- One is for running scripts
- The other is to construct packets
  - Start with this one

# What You Should See

The screenshot displays a network analysis tool interface. At the top left, a window titled "ISO Layers" contains a list with "iso-1" selected. Below this window are two buttons: "Add iso/iso options" and "Delete last layer". To the right, a "Data Link Type" dropdown menu is set to "1: ethernet [iso]". Below these controls, a "Select packet action type:" section has two radio buttons: "Send packet" (which is selected) and "Match packet". The main area of the interface is a large text box labeled "Packet details Hexadecimal / ASCII" which is currently empty. At the bottom of the interface, a status bar displays the text: "Packet data length : 000000 bytes, Adding new iso will call : iso-2 Ethernet".

## Ethernet Layer

- Click on "Add iso/iso options" to add the ethernet layer
- Notice the MAC address fields are in two parts
  - Vendor code
  - Address (remaining bytes)
- Select "hdw-from-iso3" so the correct hardware address will be filled in depending on the IP address
  - Don't use a broadcast address or you will interfere with other student experiments

# Ethernet Layer

| ISO Layers     |  |
|----------------|--|
| iso-1          |  |
| iso-2 Ethernet |  |

|               |                                         |                |
|---------------|-----------------------------------------|----------------|
| Dst vendor    | @hdw-from-iso3: match iso-3 src.address | @hdw-from-iso3 |
| Dst Address   | @hdw-from-iso3: match iso-3 src.addr    | @hdw-from-iso3 |
| Src vendor    | @hdw-from-iso3: match iso-3 dst.addr    | @hdw-from-iso3 |
| Src Address   | @hdw-from-iso3: match iso-3 dst.addr    | @hdw-from-iso3 |
| Protocol type | 0x0800: IP [iso]                        | 0x0800         |

Add iso/iso options    Delete last layer    Select packet action type:  Send packet  Match packet

Packet details Hexadecimal / ASCII

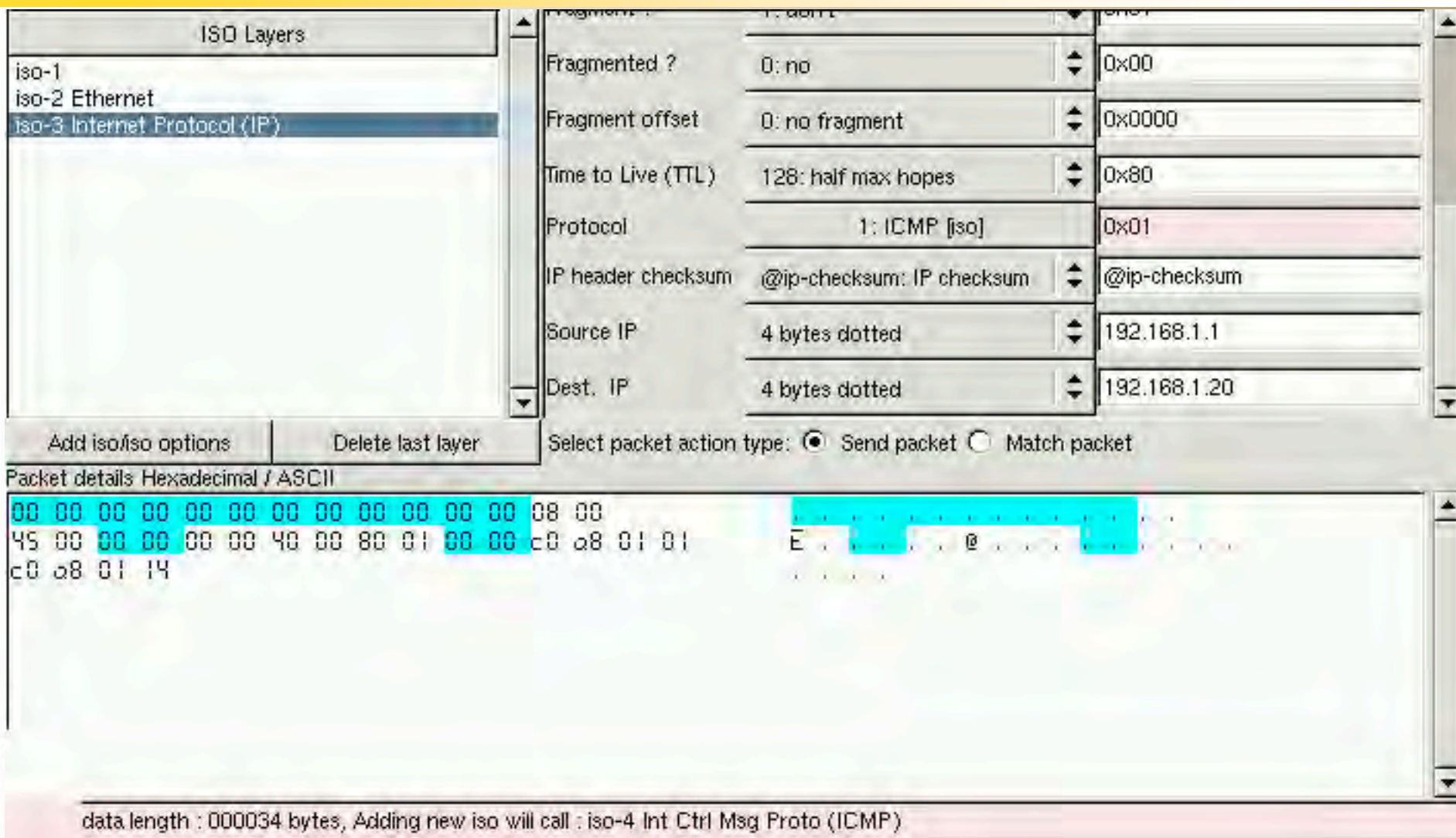
```
00 00 00 00 00 00 00 00 00 00 00 00 08 00
```

Packet data length : 000014 bytes, Adding new iso will call : iso-3 Internet Protocol (IP)

## IP Layer

- Click on "Add iso/iso options" to add the IP layer
- Scroll through the list of header fields
- Change the encapsulated protocol to "ICMP"
- Change the source IP and destination IP to those of the server and victim
  - This is supposedly a message from the gateway (192.168.1.1) to the victim (.20) to say that the server is unreachable

# IP Layer Screenshot



The screenshot displays a network configuration interface with the following components:

- ISO Layers:** A list on the left showing 'iso-1', 'iso-2 Ethernet', and 'iso-3 Internet Protocol (IP)' selected.
- Configuration Fields:**
  - Fragmented?: 0: no
  - Fragment offset: 0: no fragment
  - Time to Live (TTL): 128: half max hops
  - Protocol: 1: ICMP [iso]
  - IP header checksum: @ip-checksum: IP checksum
  - Source IP: 4 bytes dotted (192.168.1.1)
  - Dest. IP: 4 bytes dotted (192.168.1.20)
- Packet details Hexadecimal / ASCII:**

```

00 00 00 00 00 00 00 00 00 00 00 00 08 00
45 00 00 00 00 00 40 00 80 01 00 00 c0 a8 01 01
c0 a8 01 14

```
- Buttons:** 'Add iso/iso options', 'Delete last layer', and 'Select packet action type:  Send packet  Match packet'.
- Status Bar:** 'data length : 000034 bytes, Adding new iso will call : iso-4 Int Ctrl Msg Proto (ICMP)'

## ICMP Layer

- Click on "Add iso/iso options" to add the ICMP layer
- Select the ICMP type and code
  - See screenshot on next slide
- Click on "Add iso/iso options" to add the ICMP option header

# ICMP Layer Screenshot

ISO Layers

- iso-1
- iso-2 Ethernet
- iso-3 Internet Protocol (IP)
- iso-4 Int. Ctrl. Msg. Proto. (ICMP)

Type & Code: 0x0301: err: host unreachable [opt]

Checksum: @icmp-checksum: ICMP checksum

Select packet action type:  Send packet  Match packet

Packet details Hexadecimal / ASCII

```

00 00 00 00 00 00 00 00 00 00 00 00 08 00
45 00 00 00 00 00 40 00 80 01 00 00 c0 a8 01 01
c0 a8 01 14
03 01 00 00

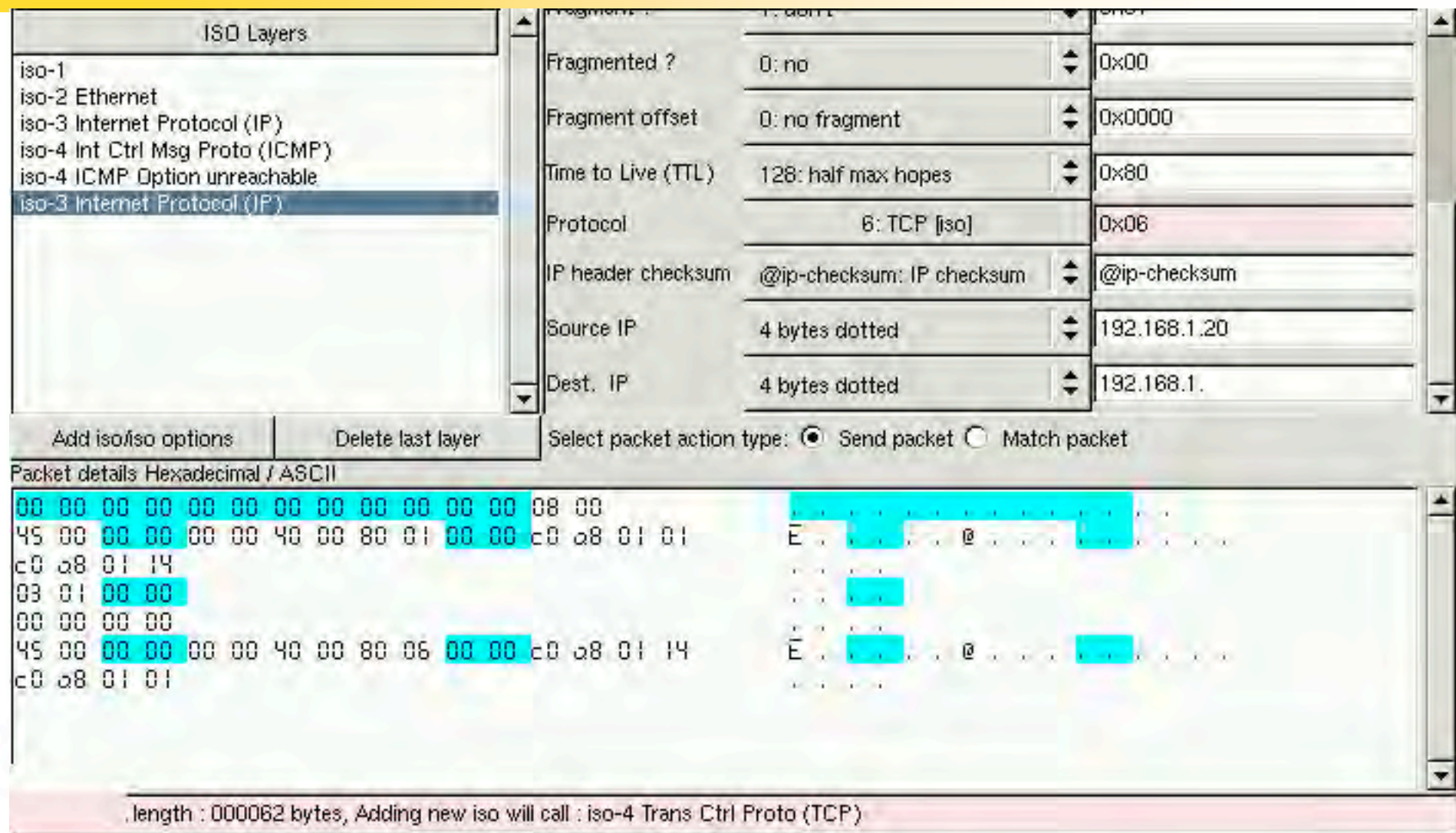
```

length : 000038 bytes, Adding new iso will call : iso-4 ICMP Option unreachable

## Add the ICMP Erroneous Data

- Click on "Add iso/iso options" to add the ICMP erroneous data
  - You should get an IP header
- Change the IP fields to appropriate values
  - Source IP
    - ❖ The victim
  - Destination IP
    - ❖ The server (not the gateway)
      - Note that the screenshot had the last IP digit erased
        - » Instructor is to provide the IP address of server

# Erroneous Data Screenshot



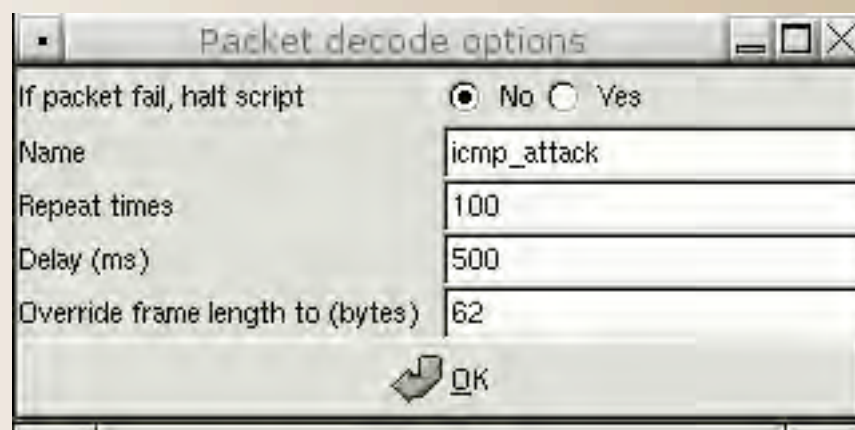
The screenshot shows a network configuration interface with the following sections:

- ISO Layers:** A list of protocol layers including iso-1, iso-2 Ethernet, iso-3 Internet Protocol (IP), iso-4 Int Ctrl Msg Proto (ICMP), iso-4 ICMP Option unreachable, and iso-3 Internet Protocol (IP) (highlighted).
- Packet Details:** A table of fields for the selected layer:
 

|                    |                           |              |
|--------------------|---------------------------|--------------|
| Fragmented ?       | 0: no                     | 0x00         |
| Fragment offset    | 0: no fragment            | 0x0000       |
| Time to Live (TTL) | 128: half max hopes       | 0x80         |
| Protocol           | 6: TCP [iso]              | 0x06         |
| IP header checksum | @ip-checksum: IP checksum | @ip-checksum |
| Source IP          | 4 bytes dotted            | 192.168.1.20 |
| Dest. IP           | 4 bytes dotted            | 192.168.1.   |
- Packet details Hexadecimal / ASCII:** A hex dump showing data bytes. Several bytes are highlighted in cyan, including 00 00, 08 00, c0 a8 01 01, 03 01 00 00, 00 00 00 00, 06 00 00 00, and c0 a8 01 01. The ASCII column shows some characters like 'E' and '@'.
- Footer:** A pink bar at the bottom contains the text: ".length : 000062 bytes, Adding new iso will call : iso-4 Trans Ctrl Proto (TCP)".

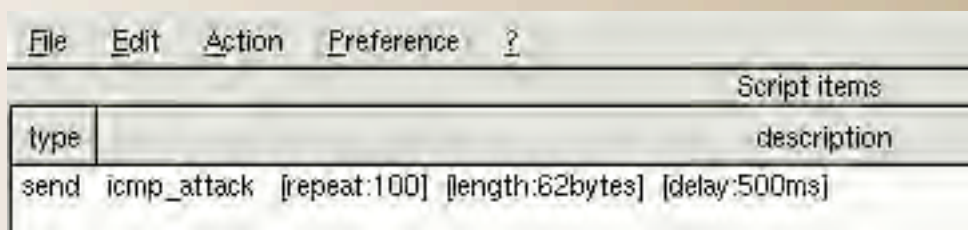
## Make a Script

- Under "Edit" select "append to script"
  - You will be creating a script
  - Experiment with the values for:
    - ❖ Number of times to send the packet
    - ❖ At which intervals



## Executing the Script

- Change to the other excalibur window
- You should see something like the screenshot below
- Click "Action", "run script" to send the packets



| File |             | Edit         |                  | Action        |  | Preference |  | ? |  |
|------|-------------|--------------|------------------|---------------|--|------------|--|---|--|
| type |             | Script items |                  |               |  |            |  |   |  |
|      |             | description  |                  |               |  |            |  |   |  |
| send | icmp_attack | [repeat:100] | [length:62bytes] | [delay:500ms] |  |            |  |   |  |

# Sniffing

- To monitor packets sent and received
  - Sender and "partner victim"
- Use tcpdump
  - You can also use ethereal or another sniffer you prefer
- Start a root terminal (right-click, select tcp tools)
- Type "tcpdump -c 20 -e -i eth0"
- This terminal will then show traffic to your station

# Testing Connectivity

- The instructor will specify a server to contact
  - SSH:
    - ❖ Account name and password to use
      - For the purposes of this lab, these could all be the same
      - Pretend that you are an attacker wanting to prevent system admins from using their accounts temporarily
        - » Or you can think of another scenario for wanting to prevent other people from using a host

## Step 4: Run the Script

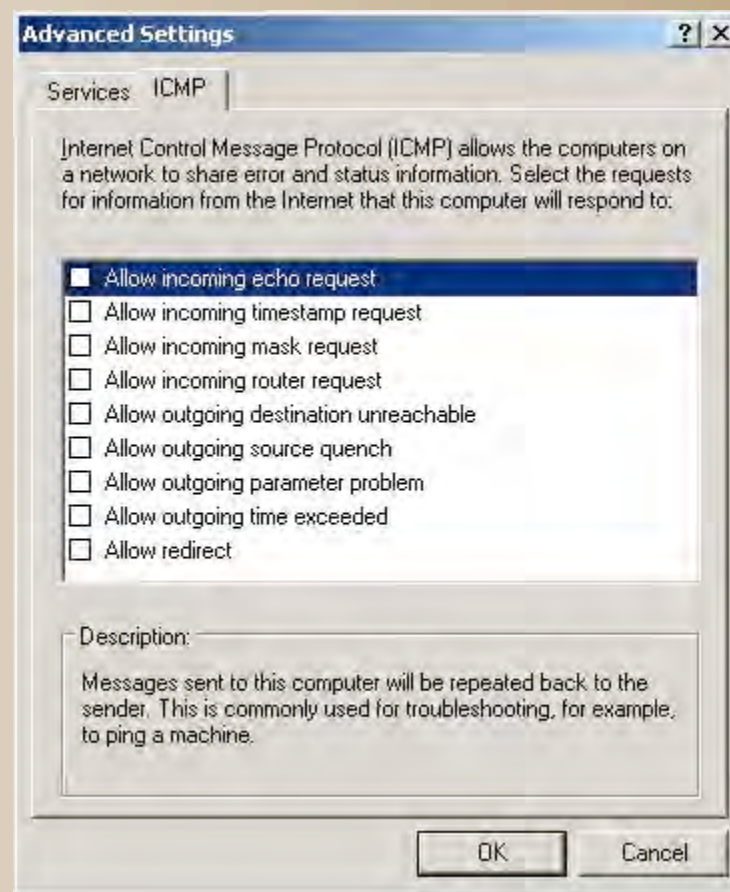
- While running the script, the "victim" should see ICMP messages being logged by tcpdump
  - If you selected carefully the values at each layer, the origin of the packets is untraceable
  - "Victim" should see tcpdump reporting things like
    - ❖ `"IP 192.168.1.1 > 192.168.1.16: icmp 48: host a.x.y.z unreachable"`
      - where a,x,y and z are the IP address of the server
- The victim should be unable to access the server, even for a while after the script is done running

## Aftermath: Discussion

- If you are configuring a firewall for a server, should you let ICMP unreachable messages through?
  - How about a firewall for an individual station?
  - How about logging the messages (and perhaps displaying an alert, or just logging an alert) without acting upon them?
- Can you trust ICMP messages from a host you trust (e.g., the network's gateway)?

# Windows XP Firewall ICMP Options

- Observe that some packet types can't be blocked



## Typical Answers

- If you are configuring a firewall for a server, should you let ICMP unreachable messages through?
  - **A server typically answers queries, so should never need to honor ICMP unreachable messages**
    - ❖ **Except to support path MTU discovery**
- How about a firewall for an individual station?
  - **It depends on whether the station is on a hostile or trusted network**
  - **When is an individual station a server?**

## Typical Answers

- How about logging the messages (and perhaps displaying an alert, or just logging an alert) without acting upon them?
  - Perhaps, as long as the logs don't obscure other events
- Can you trust ICMP messages from a host you trust (e.g., the network's gateway)?
  - No, the source IP address could have been spoofed

## ICMP Source Quench

- Supposedly sent by a router to politely ask that packets be sent more slowly
- Send many to an important host, and a sufficient slowdown may be close to a denial of service
- Might indicate to a denial-of-service attacker how effective his/her efforts are
- Really a troubleshooting message -- block it unless you are troubleshooting

## ICMP Redirect

- Type 5
- “Really, you should send your packets to that gateway first, it will be faster”  
“Uh, OK, I’ll send them there then if you say so”
- Used as a scam to perpetrate man-in-the-middle attack or DoS
- CAN-1999-1254 (under review)
  - Windows 95, 98, and NT 4.0 allow remote attackers to cause a denial of service by spoofing ICMP redirect messages from a router, which causes Windows to change its routing tables.
- Similar ideologically to ARP poisoning

# Scripted Attacks

- Winfreeze
  - Windows
  - ICMP Redirect: YOU are the quickest link to host Z
  - Host changes its routing table for Z to itself
  - Host sends packets to itself in an infinite loop
- Do you really need ICMP Redirect?
  - Do you need to act on it blindly and automatically?
    - ❖ How about asking an operator (user, administrator, root)?
  - Remember this for the discussion of routing in a later unit

# ICMP Timestamp

- Types 13 (message) and 14 (reply)
- Millisecond resolution
- Why is it bad if an attacker knows the exact time on one of your hosts?
  - Think about random number generators, for one thing
  - Can also be used to map networks, like ICMP echo (ping)
    - see “Icmpenum” (razor.bindview.com)
  - CAN-1999-0524
- NTP (network time protocol) more effective and adopted
  - ICMP messages are obsolete

# Timestamp Attack

- Pseudo-random number generators often have code like this:
- ```
// seed random number generator  
srand((double) microtime() * 1000000);
```
- So if you know the time, you know the “random” numbers.
 - May have to guess microseconds

Information Request/Reply

- Types 15 and 16
- This message is a way for a host to find out the number of the network it is on
 - a.k.a. the subnet, network address
 - Redundant if DHCP is available
- Can be used to map networks, like ICMP echo (ping) -- see “Icmpenum” (razor.bindview.com)
- Firewall suggestion: these types of packets should not be allowed through

ICMP Address Mask

- Types 17 and 18
- Used for mapping and finding info about the network
 - e.g., broadcast address to launch a smurf attack
- RFC 950
- CAN-1999-0524 (under review)
 - ICMP information such as netmask and timestamp is allowed from arbitrary hosts.

Router Solicitation and Advertisement

- Types 9 and 10
 - RFC 1256
- Trust me, I'm a gateway!
- CVE-1999-0875
 - DHCP clients with ICMP Router Discovery Protocol (IRDP) enabled allow remote attackers to modify their default routes.
- Network mapping and discovery, very useful for attackers
- See routing slides

ICMP Time Exceeded

- Type 11
- Used for traceroute
 - Allows finding gateways in routes
 - ❖ Map networks
- More useful for attackers than legitimate users
 - Firewall suggestion to prevent your network topology from being mapped:
 - ❖ Allow incoming packets (from outside the network)
 - ❖ Don't create or let them leave your network

What About MTU Discovery?

- Idea: send packets of the right size to avoid fragmentation
- Set “don’t fragment” bit
- Listen for Destination Unreachable ICMP packets (code 4 means “fragmentation needed”)
- Performance tweak that is not strictly necessary
 - Always on in IPv6
- How can you abuse it?

Messing up PMTU

- CAN-2001-0323
 - The ICMP path MTU (PMTU) discovery feature in various UNIX systems allows remote attackers to cause a denial of service by spoofing "ICMP Fragmentation needed but Don't Fragment (DF) set" packets between two target hosts, which could cause one host to lower its MTU when transmitting to the other host.
- IPv4 Minimum MTU 68 bytes
 - Is it correctly implemented?
 - Set the MTU to 21 bytes
 - ❖ Transmission overhead: 20 times the payload!
 - ❖ What happens if the MTU is set to 20?

What to do?

- Routers and firewalls need to send ICMP messages to support path MTU discovery
 - Otherwise packets fall into a black hole
 - Alternative: don't honor the "don't fragment" bit and silently fragment
- Should you use PMTU and honor ICMP messages?
 - The minimum MTU is configurable (e.g., with DHCP)
- IPv6
 - No "don't fragment" bit (always on)
 - The minimum MTU size has been increased to 1280 bytes
 - ❖ Less attractive target
 - Routers don't fragment packets anymore

Example ICMP Attack Tool

- SING
 - Stands for 'Send ICMP Nasty Garbage'
 - Sends ICMP packets fully customized from command line
 - Replaces the ping command but adds certain enhancements (Fragmentation, spoofing,...)
 - <http://sourceforge.net/projects/sing/>

ICMP-related Vulnerabilities

- Denial of service by sending forged ICMP unreachable packets.
 - CVE-1999-0214
- Denial of service in Gauntlet Firewall via a malformed ICMP packet.
 - CVE-1999-0683
- Denial of service in Linux 2.2.x kernels via malformed ICMP packets containing unusual types, codes, and IP header lengths.
 - CVE-1999-0804

ICMP Problems (cont.)

- ICMP messages to broadcast addresses are allowed, allowing for a Smurf attack that can cause a denial of service.
 - CVE-1999-0513
- Jolt ICMP attack causes a denial of service in Windows 95 and Windows NT systems.
 - CAN-1999-0345 (under review)
- Linux kernel, and possibly other operating systems, allows remote attackers to read portions of memory via a series of fragmented ICMP packets that generate an ICMP TTL Exceeded response, which includes portions of the memory in the response
 - CVE-2002-0046

ICMP Problems (cont.)

- Sybergen Secure Desktop 2.1 does not properly protect against false router advertisements (ICMP type 9), which allows remote attackers to modify default routes.
 - CVE-2000-0568
- Cisco 12000 with IOS 12.0 and line cards based on Engine 2 and earlier allows remote attackers to cause a denial of service (CPU consumption) by flooding the router with traffic that generates a large number of ICMP Unreachable replies.
 - CVE-2001-0861

ICMP Problems (cont.)

- DHCP clients with ICMP Router Discovery Protocol (IRDP) enabled allow remote attackers to modify their default routes.
 - CVE-1999-0875
- Reliant Unix 5.44 and earlier allows remote attackers to cause a denial of service via an ICMP port unreachable packet, which causes Reliant to drop all connections to the source address of the packet.
 - CAN-2001-0411

Conclusions

- You don't need most of ICMP unless you need to troubleshoot your network
- ICMP is very useful to attackers, rarely useful to legitimate users.
 - Except Path MTU discovery
 - e.g., OS fingerprinting
- Blocking ICMP by default in critical networks, and logging ICMP messages instead of acting upon them automatically, is safer

Reference

- ICMP Usage in Scanning: The Complete Know-How
 - Ofir Arkin 2001
- Xprobe: fingerprinting with ICMP (Yarochkin et al. 2005)
- Interesting conclusions
 - all types of ICMP packets can be used for scanning
 - ICMP packets can be used for:
 - ❖ DoS attacks
 - ❖ DDoS attacks
 - ❖ Covert channel communications
 - e.g., controlling zombies without detection

Inverse Mapping

- Idea: send packets (any protocol, e.g., ICMP echo reply works) to scan a network. Invalid IPs will generate ICMP host unreachable messages.
- ICMP Unreachable messages form a "negative" image of the network.
 - IP addresses for which you didn't get unreachable messages are likely to be valid ones.
- Does your router need to respond with ICMP unreachable messages?

Question

- What is the best strategy against Smurf attacks?
 - a) block incoming pings to broadcast addresses at the firewall or router
 - b) complain to the vendor, asking for a patch
 - c) install an IDS
 - d) block all ICMP echo traffic at the firewall or router
 - e) strike back at the attacker by spoofing her IP address in another Smurf

Question

- What is the best strategy against Smurf attacks?
 - a) **block incoming pings to broadcast addresses at the firewall or router (Good idea:low cost, high effectiveness)**
 - b) complain to the vendor, asking for a patch (probably won't change anything)
 - c) install an IDS (an Intrusion Detection System will let you know that you were attacked, but won't prevent it)
 - d) block all ICMP echo traffic at the firewall or router (blocks legitimate pings)
 - e) strike back at the attacker by spoofing her IP address in another Smurf (bad idea)

Question

- Name two in this list that can't be used to scan a network (all by themselves):
 - a) echo request
 - b) echo reply
 - c) timestamp request
 - d) network mask request
 - e) source quench

Question

- Name two in this list that can't be used to scan a network (all by themselves):
 - a) echo request
 - b) echo reply**
 - c) timestamp request
 - d) network mask request
 - e) source quench**
- **Choices b) and e) don't initiate responses for valid packets, so don't reveal the presence of hosts. However, a router *might* respond with a host unreachable message, enabling inverse mapping! (c.f. Ofir Arkin's paper)**

Question

- What do you break if you disallow ICMP unreachable messages?
 - a) MTU discovery
 - b) TraceRoute
 - c) DNS
 - d) ARP
 - e) Nothing

Question

- What do you break if you disallow ICMP unreachable messages?
 - a) **MTU discovery (uses Unreachable ICMP messages, subtype fragmentation needed but DF set)**
 - b) TraceRoute (uses Time Exceeded ICMP messages)
 - c) DNS (the Domain Name System has nothing to do with ICMP)
 - d) ARP (ARP has nothing to do with ICMP)
 - e) Nothing

Discussion

- What is the fundamental problem with ICMP?
- Can you suggest likely firewall rules relating to ICMP?

Question

- If you receive an ICMP network redirect message, which attack could possibly be attempted against you?
 - a) Man-in-the-middle
 - b) WinFreeze
 - c) DoS
 - d) a and c
 - e) all of the above

Question

- If you receive an ICMP network redirect message, which attack could possibly be attempted against you?
 - a) Man-in-the-middle (could be, for intercepting messages)
 - b) WinFreeze (lots of ICMP host redirects, spoofed from gateway)
 - c) DoS (could be that packets are sent to a black hole)
 - d) a and c**
 - e) all of the above if they are host redirect

Question

- If you receive an ICMP unreachable message with the flags “Fragmentation needed but Don't Fragment (DF) set”, which attack could be attempted against you?
 - a) Partial DoS
 - b) Jolt
 - c) Smurf
 - d) WinFreeze
 - e) Ping of Death

Question

- If you receive an ICMP unreachable message with the flags “Fragmentation needed but Don't Fragment (DF) set”, which attack could be attempted against you?
 - a) Partial DoS (someone may be trying to mess with MTU discovery)**
 - b) Jolt (identical fragmented packets)
 - c) Smurf (ping to a broadcast address)
 - d) WinFreeze (ICMP host redirect spoofed from gateway)
 - e) Ping of Death (fragmented ping that can't be reassembled)

IGMP v.2

- Internet Group Management Protocol
- For multicast messages
 - Remember multicast MAC addresses?
- Multicast routers keep a list of multicast group memberships for each attached network
- Messages are unauthenticated
 - A malicious host can
 - ❖ Get other hosts out of a group
 - ❖ Create complex data structures in the router
 - ❖ etc...
- IGMP can also be used for IP fragmentation attacks

IGAP

- IGAP adds user authentication to IGMP for joining multicast groups
- Needs trust relationship between content provider and network service provider
 - Not trivial in the general internet case
 - Could work with content owned by ISP (e.g., cable cos)
 - Routers need to query and maintain authentication information

IGMP v.3

- Adds source filtering capabilities
 - white list
 - ❖ sources whose content is desired
 - black list
 - ❖ sources whose content shouldn't be sent to me
- No authentication capabilities by itself
 - Reliance on IPSEC is suggested
 - ❖ RFC 3376
 - Proposal for "once-in-a-while" authentication ("Upload Authentication Information Using IGMPv3, <draft-he-magma-igmpv3-auth-00.txt>") not included in RFC 3376

Additional References

- Ramachandran et al. IGMP DoS Vulnerabilities (2002)
 - <http://www.securiteam.com/securitynews/5XP0B1F7FY.html>

ICMP v.6

- Works with IP v.6
- RFC 1885
- Merges IGMP and ICMP v.4
- Some messages are gone
 - source quench
 - redirect
- No authentication unless IPSEC is used
- Could allow attacks on protocols (services) otherwise protected by a firewall
 - Error messages, (including unknown types), MUST be passed up to the "protocol entity"

Discussion

- Do you know of a successful, practical application of IGMP?
 - OSPF
 - Any others?
- Do you need it?
 - Most people wouldn't notice if it didn't exist on their LAN segment
- Conclusion: IGMP is optional
 - In secure networks, if you don't need it, dispense with it

Questions or Comments?

§

About These Slides

- You are free to copy, distribute, display, and perform the work; and to make derivative works, under the following conditions.
 - You must give the original author and other contributors credit
 - The work will be used for personal or non-commercial educational uses only, and not for commercial activities and purposes
 - For any reuse or distribution, you must make clear to others the terms of use for this work
 - Derivative works must retain and be subject to the same conditions, and contain a note identifying the new contributor(s) and date of modification
 - For other uses please contact the Purdue Office of Technology Commercialization.
- Developed thanks to the support of Symantec Corporation

Pascal Meunier

pmeunier@purdue.edu

Contributors:

Jared Robinson, Alan Krassowski, Craig Ozancin, Tim Brown, Wes Higaki, Melissa Dark, Chris Clifton, Gustavo Rodriguez-Rivera

