

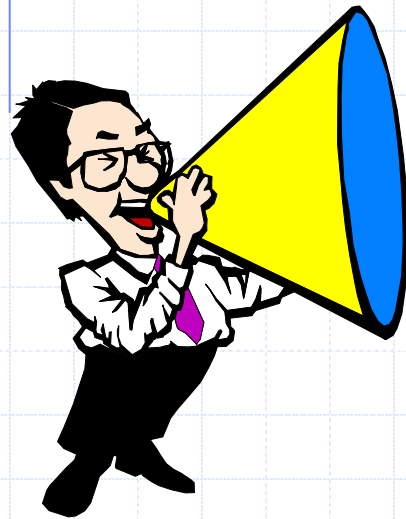
Promiscuous node detection using ARP packets

Daiji Sanai

<hyler@securityfriday.com>

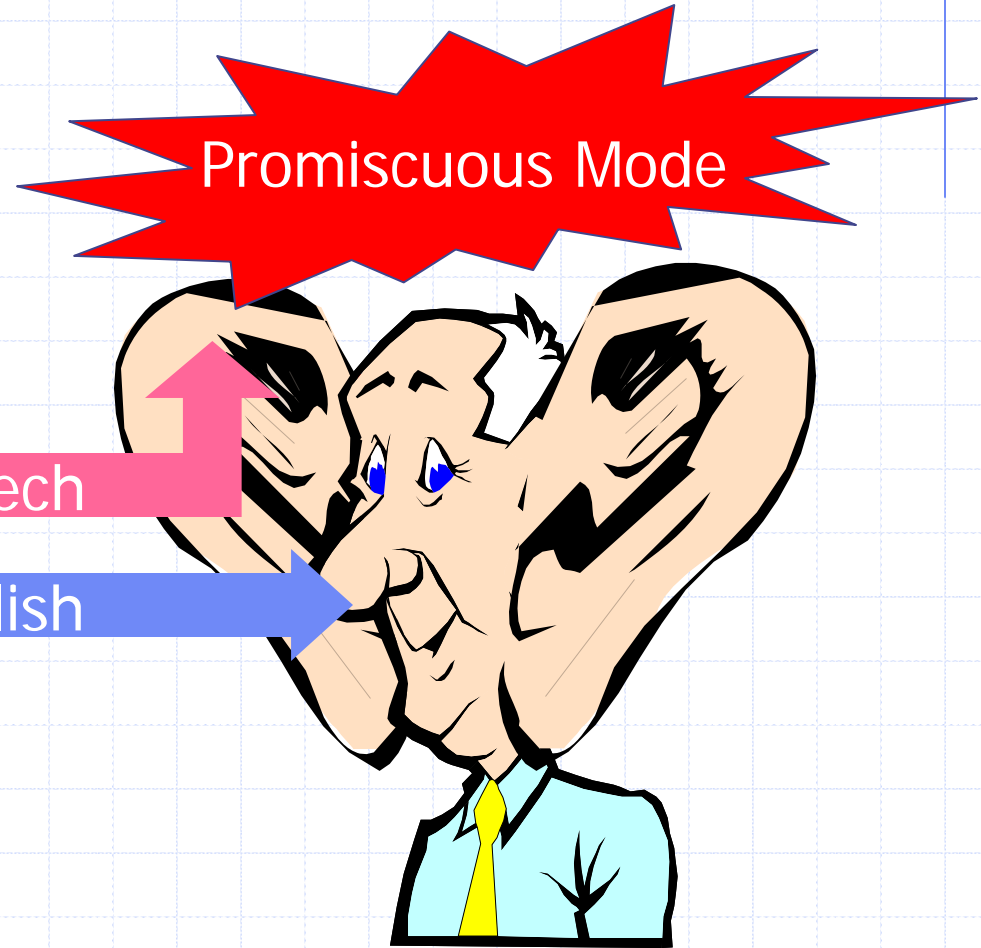
SecurityFriday.com

README.TXT



My speech

English



Agenda

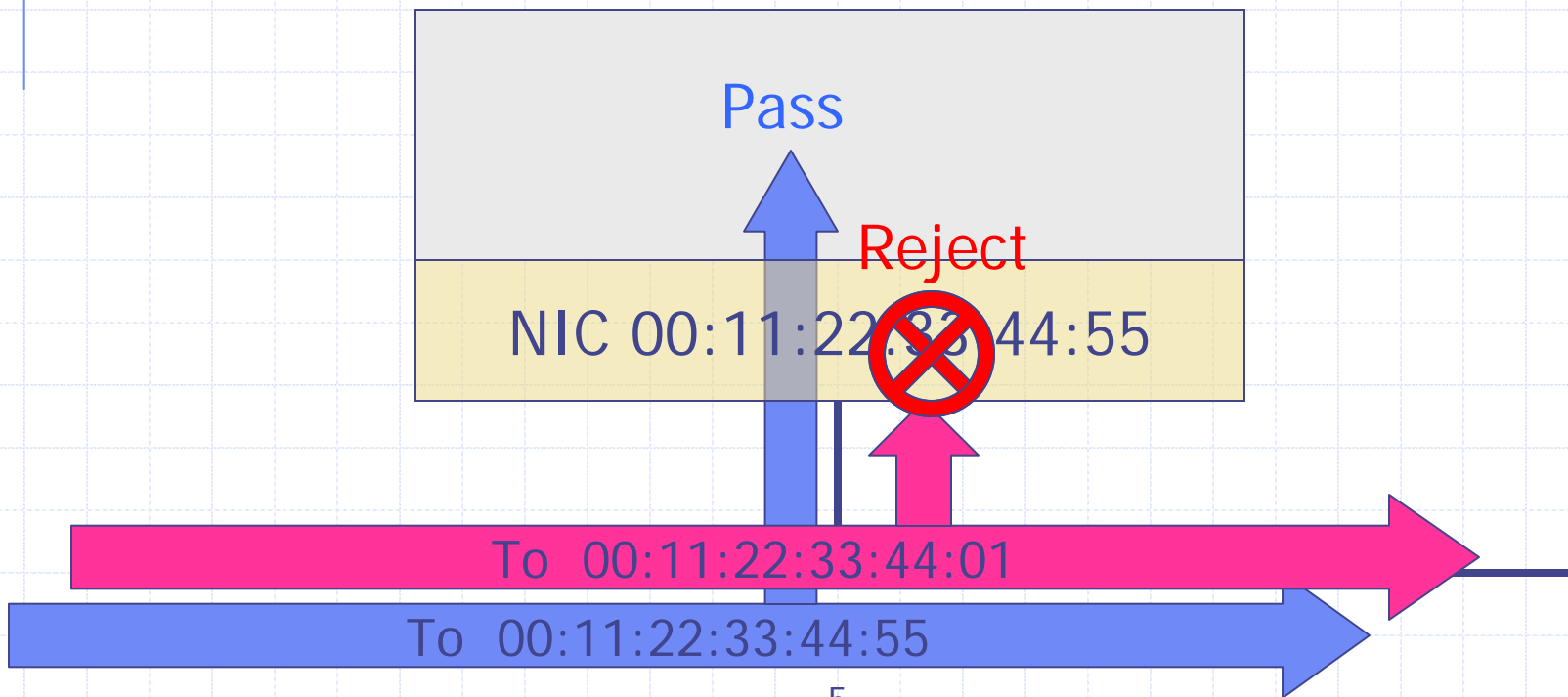
- ◆ Hardware filter
- ◆ Address Resolution Protocol
- ◆ Software filter
- ◆ Promiscuous detection
- ◆ Exception

Hardware filter

- ◆ Unicast (to host)
- ◆ Broadcast
- ◆ Multicast
- ◆ All multicast
- ◆ Promiscuous

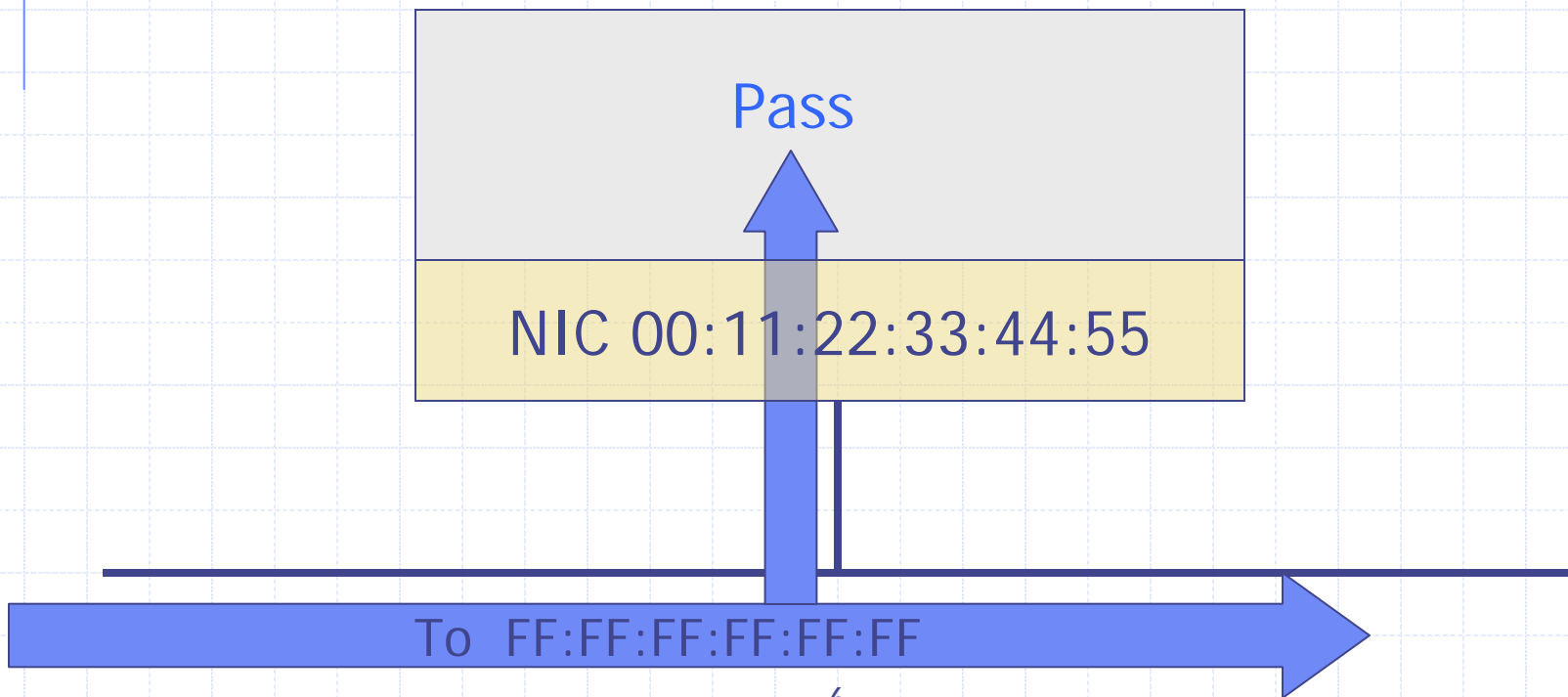
Unicast (to host)

- ◆ The packet to the HW address of the device is passed.



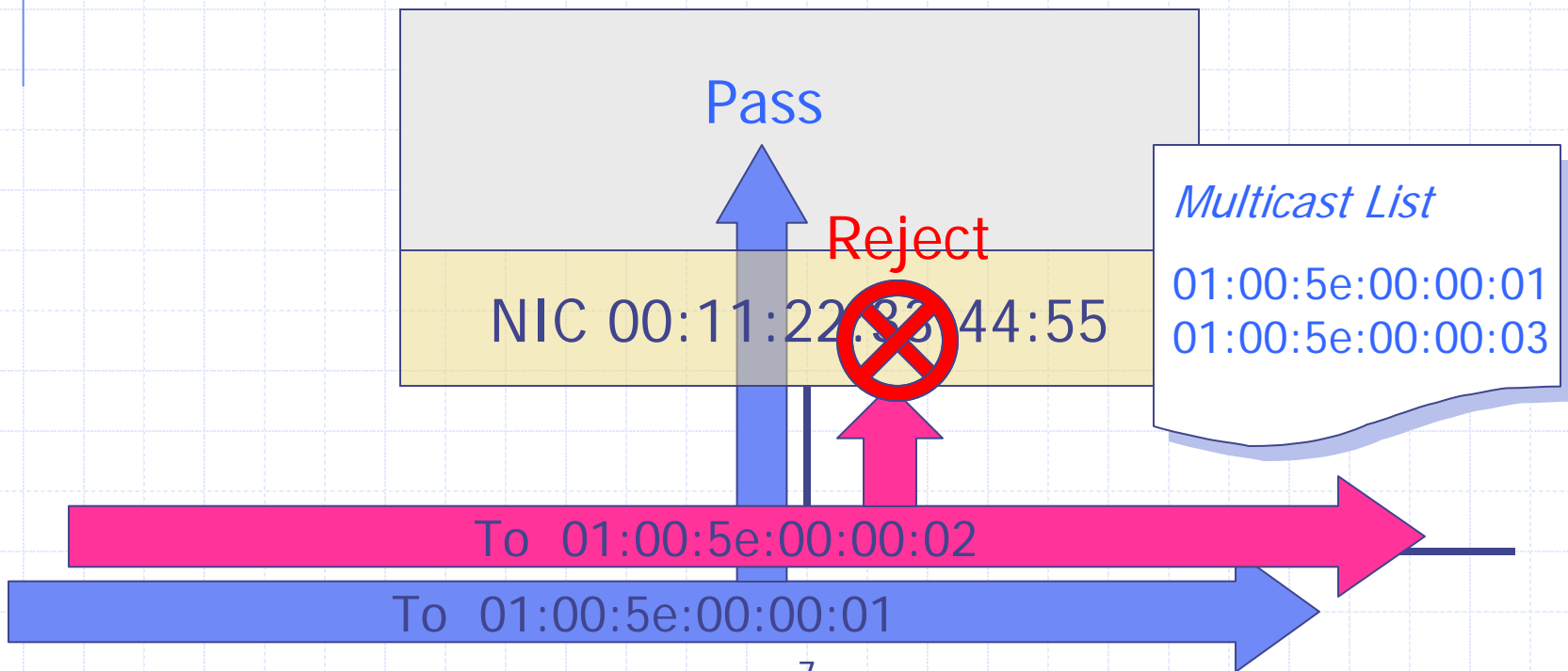
Broadcast

- ◆ Packet to broadcast (FF:FF:FF:FF:FF:FF) is passed



Multicast

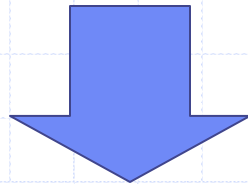
- ◆ The address registered in the multicast list is passed.



All multicast

- ◆ The multicast packet of all groups passes.

What is the multicast packet?



It is the packet where the **group bit** is set to multicast.

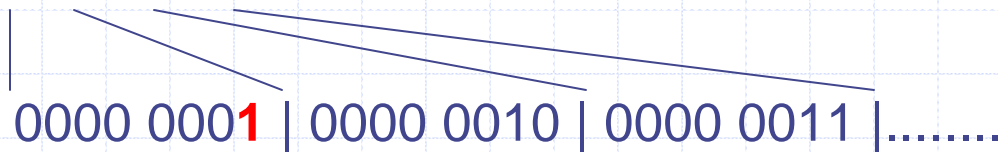
All multicast (2)

◆ The packet which sets the group bit is passed

- Group bit

HW Address:

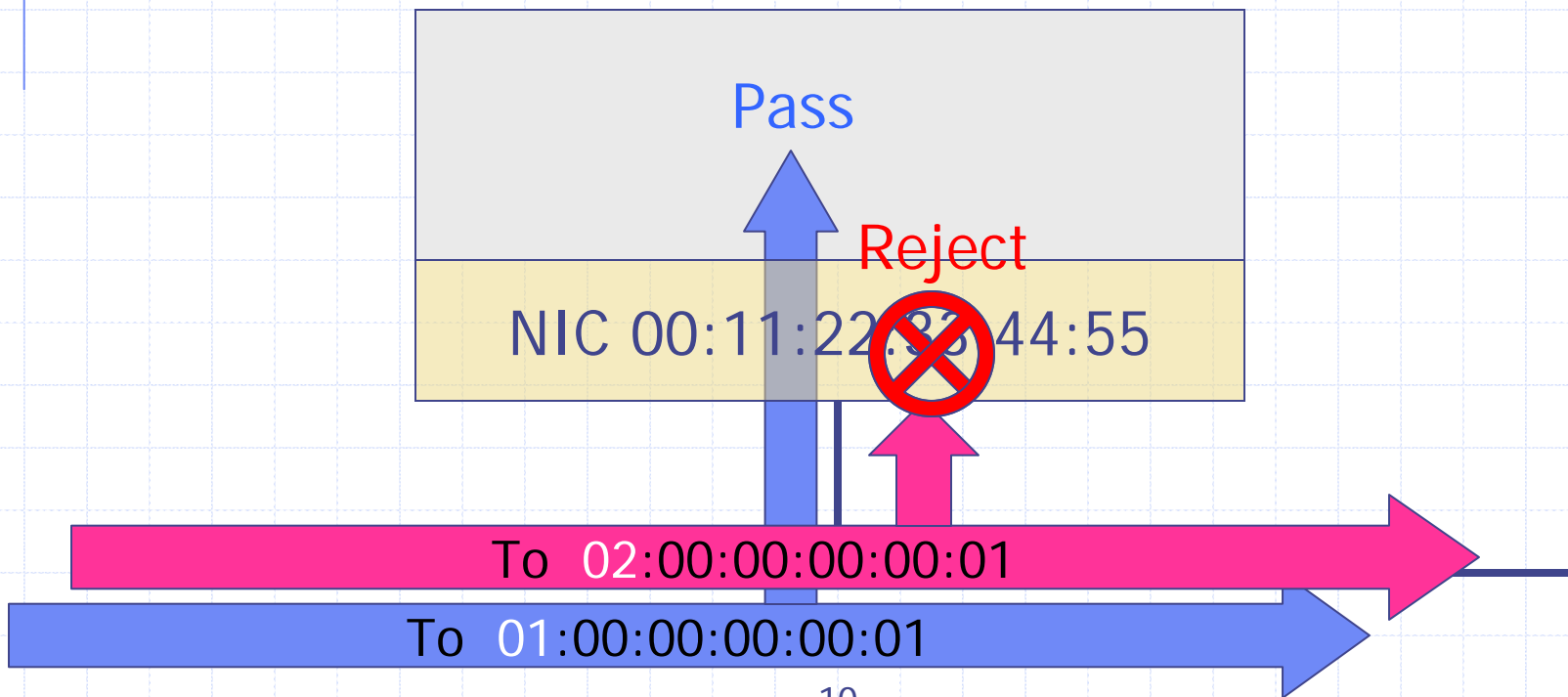
01:02:03:04:05:06



group bit

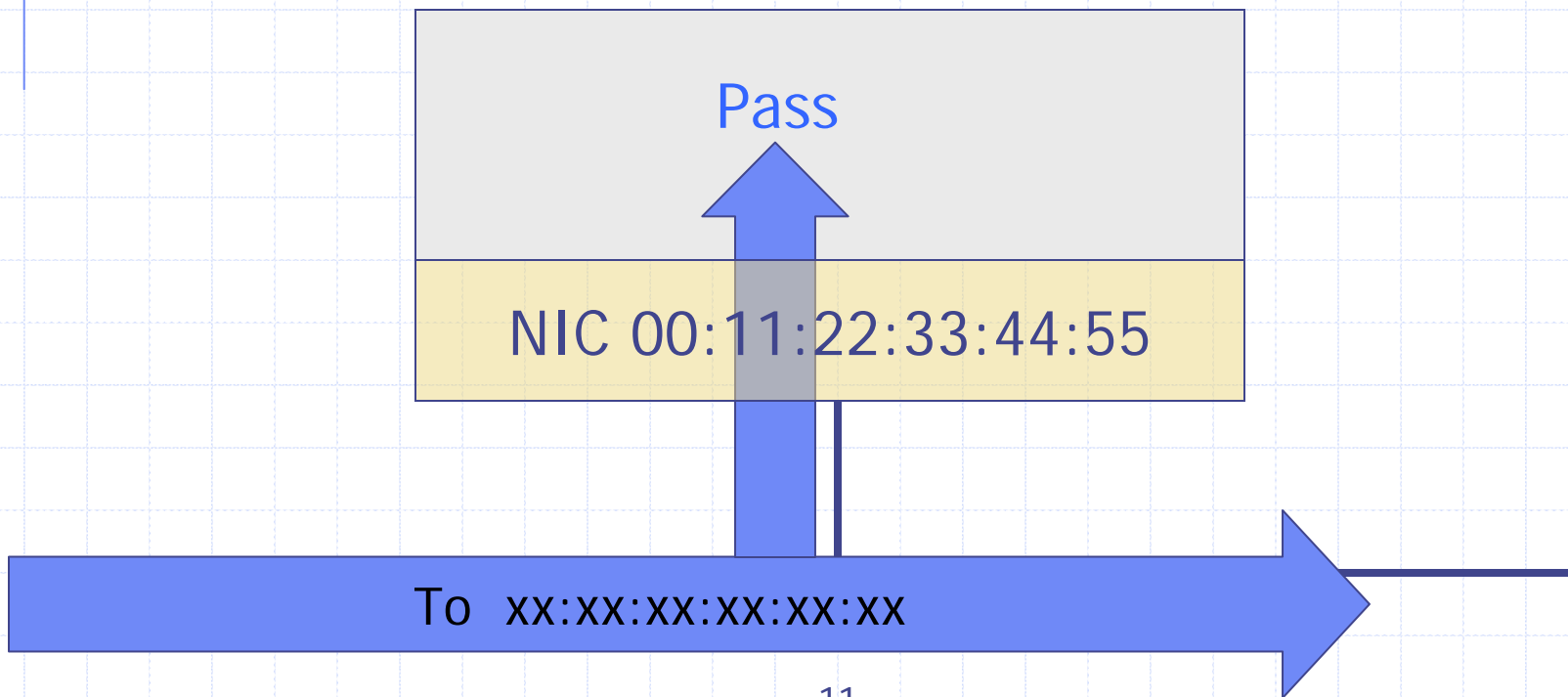
All multicast (3)

- ◆ The packet which sets the group bit is passed



Promiscuous

- ◆ All packets are passed.



Default HW filter

◆ Unicast

- HW Address
(ex. 00:11:22:33:44:55)

◆ Broadcast

- FF:FF:FF:FF:FF:FF

◆ Multicast

- Multicast address 1
01:00:5E:00:00:01

ARP

◆ Address Resolution Protocol

- Protocol to search for HW address which corresponds to IP address

ARP (2)

- ◆ Requested IP address is set in the ARP packet.
- ◆ The packet is sent to the broadcast address.
- ◆ The requested node replies with its' HW address.

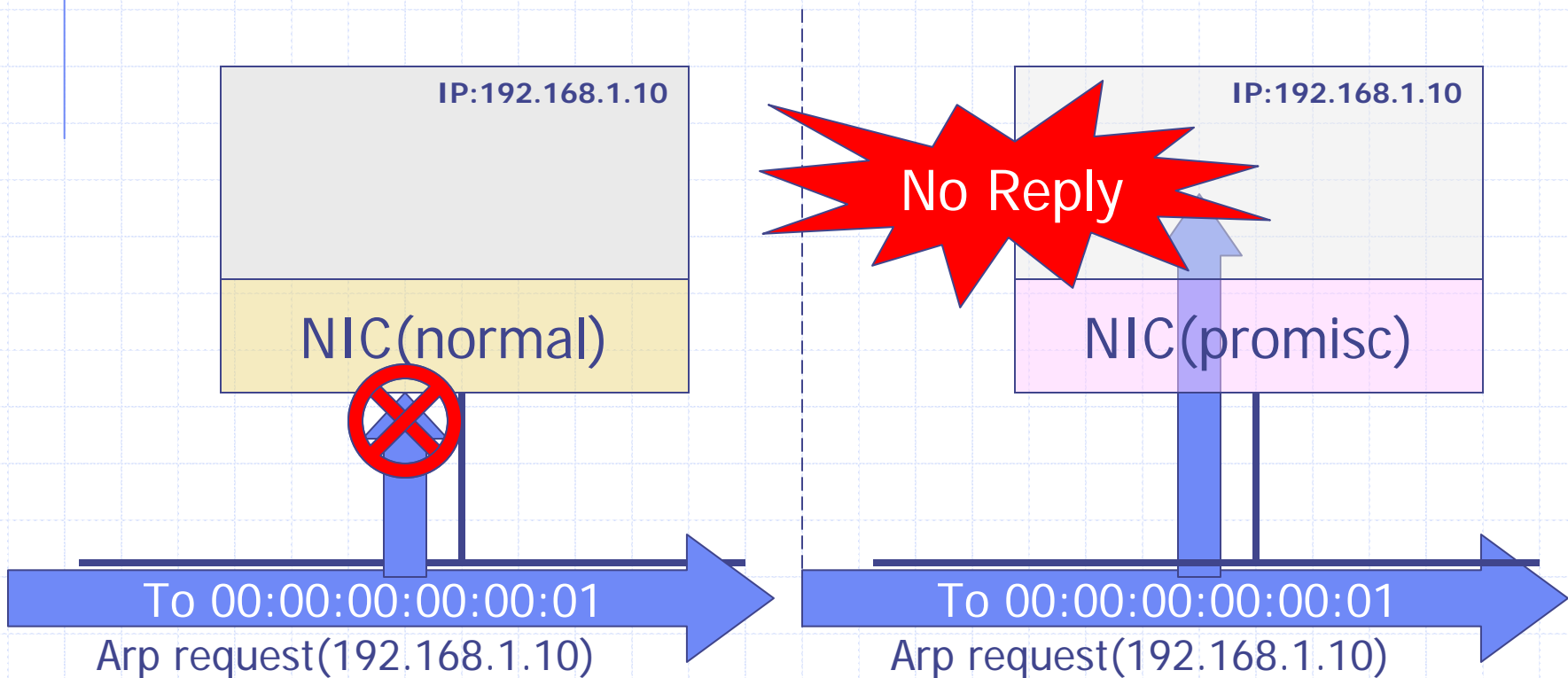
Packet format of ARP

◆ ARP packet (request)

| | | |
|---------|-------------------------------------------|-------------------------|
| 6bytes: | Ethernet address of destination | FF FF FF FF FF FF |
| 6bytes: | Ethernet address of sender | 00 11 22 33 44 55 |
| 2bytes: | Protocol type (ARP=0806) | 08 06 |
| 2bytes: | Hardware address space (ethernet=01) | 00 01 |
| 2bytes: | Protocol address space (IPv4=0800) | 08 00 |
| 1byte: | byte length of hardware address | 06 |
| 1byte: | byte length of protocol address | 04 |
| 2bytes: | opcode (arp request=01 ,arp reply=02) | 00 01 |
| 6bytes: | Hardware address of sender of this packet | 00 11 22 33 44 55 |
| 4bytes: | Protocol address of sender of this packet | <i>My IP</i> |
| 6bytes: | Hardware address of target of this packet | 00 00 00 00 00 00 |
| 4bytes: | Protocol address of target | <u><i>Target IP</i></u> |

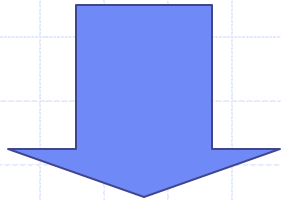
Test 1

- Does not set the broadcast address in the HW Address of the ARP Packet.



Consideration of test 1

- ◆ Why is there no reply ?
 - Something is set in the software filter.



- ◆ What kind of filter ?
 - Multicast?
 - Broadcast?

linux/arp.c (1)

```

if (in_dev == NULL ||
    arp->ar_hln != dev->addr_len  ||
    dev->flags & IFF_NOARP ||
    skb->pkt_type == PACKET_OTHERHOST ||
    skb->pkt_type == PACKET_LOOPBACK ||
    arp->ar_pln != 4)
    goto out;
//check hw addr length
//no arp
//otherhost packet
//loopback packet
//ipv4

switch (dev_type) {
default:
    if (arp->ar_pro != __constant_htons(ETH_P_IP))
        goto out;
//ip protocol 0800
    if (htons(dev_type) != arp->ar_hrd)
        goto out;
//check hw device
    break;

```

linux/arp.c (2)

```

if (arp->ar_op != __constant_htons(ARPOP_REPLY) &&           //arp
    request or reply
    arp->ar_op != __constant_htons(ARPOP_REQUEST))
    goto out;

```



```

/*
 * Check for bad requests for 127.x.x.x and requests for multicast
 * addresses. If this is one such, delete it.
 */
if (LOOPBACK(tip) || MULTICAST(tip)) //loopback or multicast
    goto out;

```

Check IP Address

linux/arp.c (3)

◆ filter of ARP module

- ARP message is correct.
- A packet is not OTHERHOST.
- A packet is not LOOPBACK.
- Request IP Address is not loopback.
- Request IP Address is not multicast.



ARP responds if the HW address of the packet is
TO_US, BROADCAST, or MULTICAST.

Classification of packet

◆ In the software

- What is a TO_US packet ?
- What is a MULTICAST packet?
- What is a BROADCAST packet?

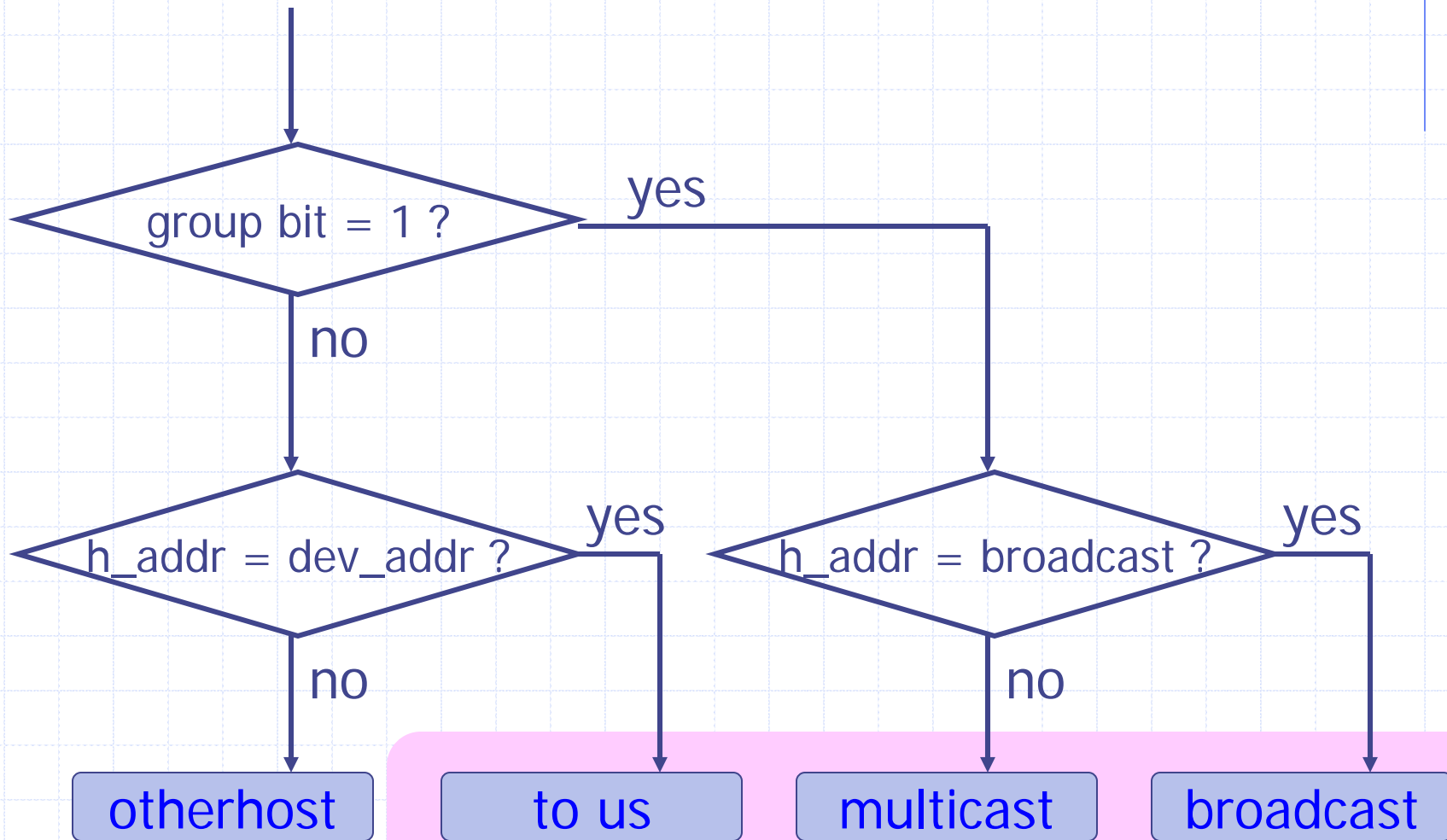
linux/eth.c (1)

```
if(*eth->h_dest&1)
{
    if(memcmp(eth->h_dest,dev->broadcast, ETH_ALEN)==0)
        skb->pkt_type=PACKET_BROADCAST;
    else
        skb->pkt_type=PACKET_MULTICAST;
}

/*
 * This ALLMULTI check should be redundant by 1.4
 * so don't forget to remove it.
 *
 * Seems, you forgot to remove it. All silly devices
 * seems to set IFF_PROMISC.
 */

else if(1 /*dev->flags&IFF_PROMISC*/)
{
    if(memcmp(eth->h_dest,dev->dev_addr, ETH_ALEN))
        skb->pkt_type=PACKET_OTHERHOST;
}
```

linux/eth.c (2)



ARP Response

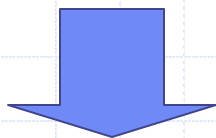
for Linux

| | gr bit | normal mode | | | promiscuous mode | | |
|--------------------------------|--------|-------------|-----------|------|------------------|-----------|------|
| | | hw filter | sw filter | res. | hw filter | sw filter | res. |
| to_us | off | → | → | ✓ | → | → | ✓ |
| other host | | reject | - | - | → | reject | - |
| broadcast | on | → | → | ✓ | → | → | ✓ |
| multicast (in the list) | | → | → | ✓ | → | → | ✓ |
| multicast (not in the list) | | reject | - | - | → | → | ✓ |
| group | | reject | - | - | → | → | ✓ |

SW filter of Windows

- ◆ I do not know.
- ◆ I have not seen the source code.

However, there is something in the filter.



Test 2

Test 2

◆ A special HW address is set and tested.

- FF:FF:FF:FF:FF:FF Broadcast
- FF:FF:FF:FF:FF:FE Fake broadcast (31bits)
- FF:FF:00:00:00:00 Fake broadcast (word)
- FF:00:00:00:00:00 Fake broadcast (byte)
- 01:00:5E:00:00:00 Multicast address 0
- 01:00:5E:00:00:01 Multicast address 1
- 01:00:00:00:00:00 Group bit

◆ OS

- Windows9x/2000,Linux

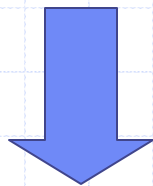
Result 2

| HW Address | Windows9x/ME | | Windows2k/NT4 | | Linux2.2/2.4 | |
|-------------------|--------------|---------|---------------|---------|--------------|---------|
| | normal | promisc | normal | promisc | normal | promisc |
| FF:FF:FF:FF:FF:FF | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| FF:FF:FF:FF:FF:FE | - | ✓ | - | ✓ | - | ✓ |
| FF:FF:00:00:00:00 | - | ✓ | - | ✓ | - | ✓ |
| FF:00:00:00:00:00 | - | ✓ | - | - | - | ✓ |
| 01:00:00:00:00:00 | - | - | - | - | - | ✓ |
| 01:00:5E:00:00:00 | - | - | - | - | - | ✓ |
| 01:00:5E:00:00:01 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Exception 1

◆ Old NIC does not support the multicast list.

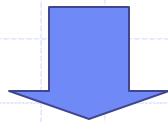
- EtherLink III etc.
 - ◆ A multicast list isn't supported.



- ◆ Default is all multicast.
 - The packet which sets the group bit is passed

Exception 2

◆ Linux+3c905 (Dell on board is the same.)
is always all multicast



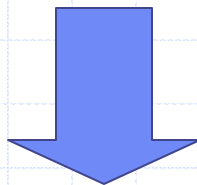
◆ The installer automatically sets it to the older
driver 3c59x.o (in which ,multicast list isn't supported.).



◆ When the newer driver ,3c90x.o, is set it is
correct.

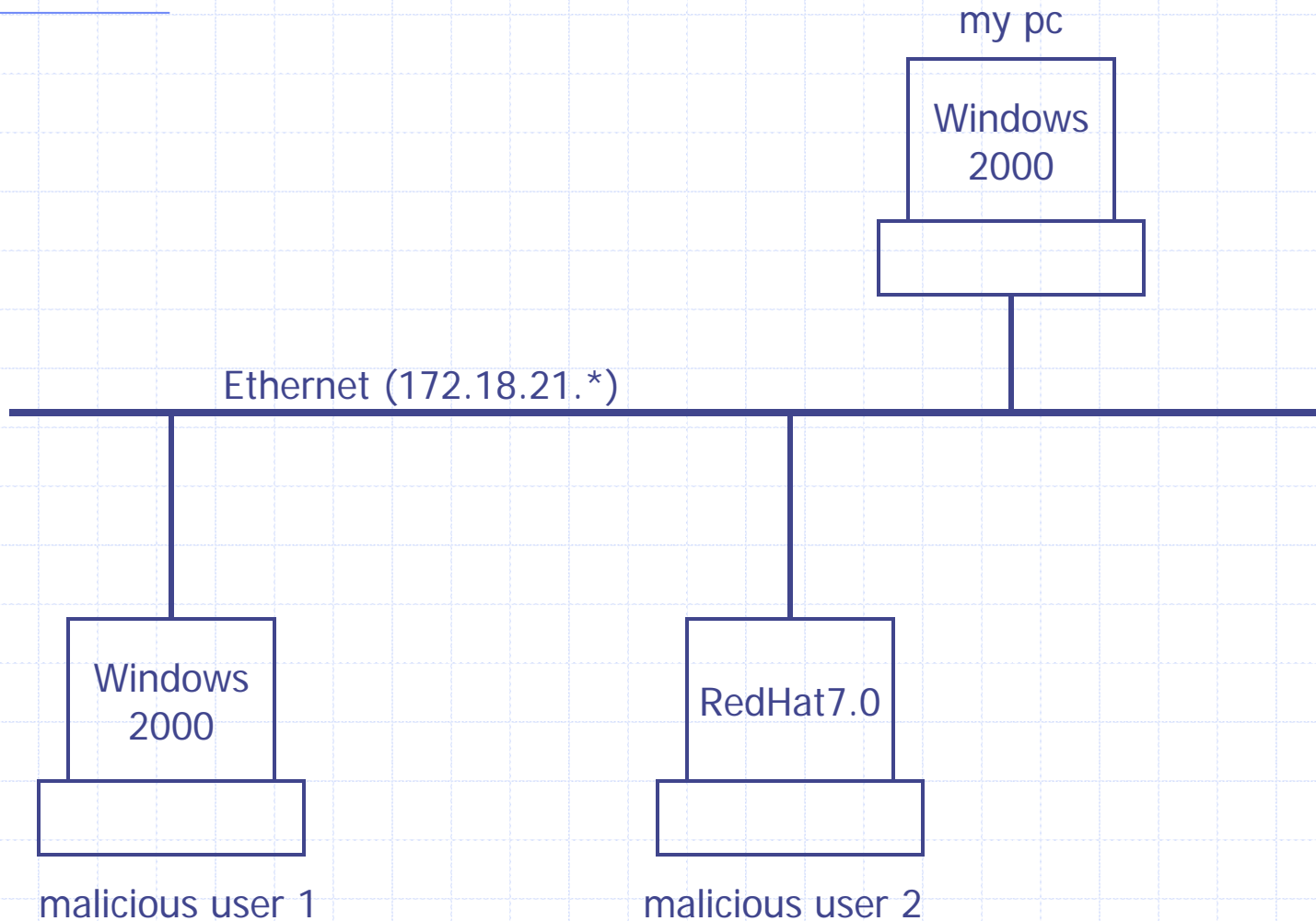
Exception 3

- ◆ Windows2000 dynamically loaded driver
 - WinPcap2.1 and SMS(Systems Management Server)



- normally responds to FF:FF:00:00:00:00.
- responds to FF:FF:FF:FF:FF:FE in promiscuous

Demonstration



Test tool

◆ You can download the test tool from our site.

- **PromiScan**

<http://www.securityfriday.com/>

Please report your test results to us.

Contact Information

- Daiji Sanai
hyler@securityfriday.com
- SecurityFriday
<http://www.securityfriday.com/>

A decorative graphic consisting of a vertical blue line on the left side, a horizontal blue line extending from the top of the vertical line, and a small blue circle at the top-left corner of the horizontal line.

Thank you