

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

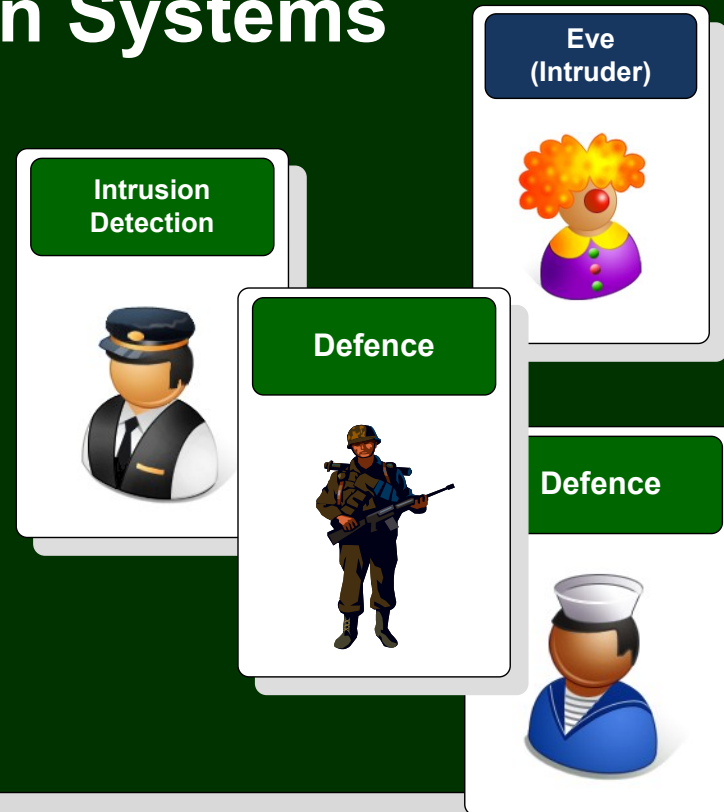
A few intrusions

User profiling

Honeypots

IPS

Conclusions



Introduction

Enemy takes some time to breach each of the levels of defence



Defence



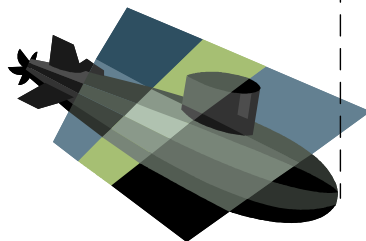
Defence



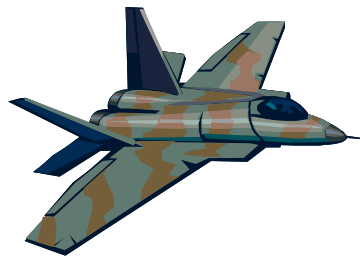
Defence



Defence



**Forth-level  
defence**



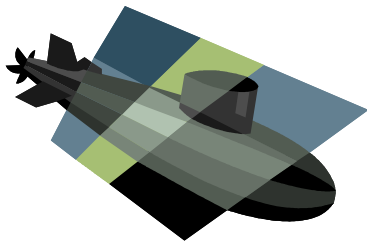
**Third-level  
defence**



**Second-level  
defence**



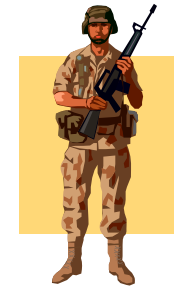
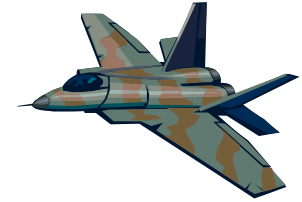
**First-level  
defence**



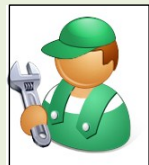
**Trusted  
(our side)**



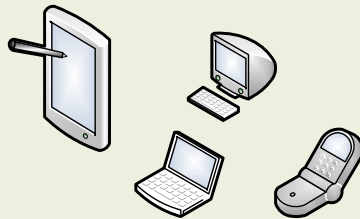
**DMZ – an area  
where military  
actions  
are prohibited**



**Untrusted  
(their side)**



Users

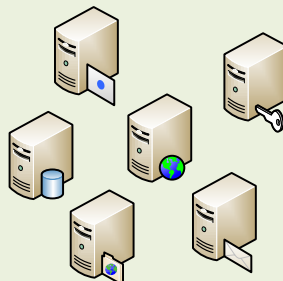


Assets



Hello. How are you? Is this okay?

Data



Systems



# Intrusion Detection Systems can help to reduce breaches

Introduction

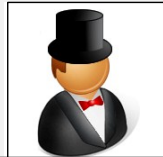
IDS



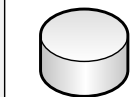
Systems



Assets



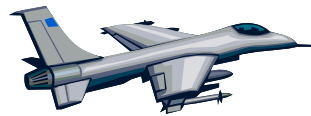
Users



Data



Forth-level  
defence



Third-level  
defence



Second-level  
defence



First-level  
defence

Defence-in-the-depth

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

Honeypots

IPS

Conclusions

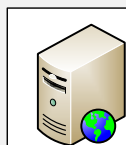


Threats

Corporate access

Email access

Web access



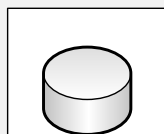
Systems



Assets



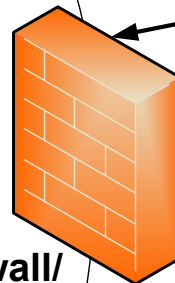
Users



Data

Network/  
Organisational  
perimeter

Firewall/  
gateway



Data  
stealing



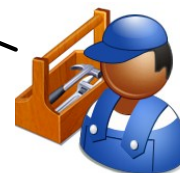
External  
hack



DoS (Denial-of-  
service)



Personal  
abuse



Worms/viruses



Terrorism/  
extortion



Fraud

**CSI (Computer Security Institute) found:**

- 70% of organisation had breaches
- 60% of all breaches came from inside their own systems

Corporate access

Email access

Web access



Systems



Assets

Users

Data

Firewall/  
Gateway

(cannot deal with  
internal threats)

Data  
stealing

External  
hack

DoS (Denial-of-  
service)

Personal  
abuse

Worms/viruses

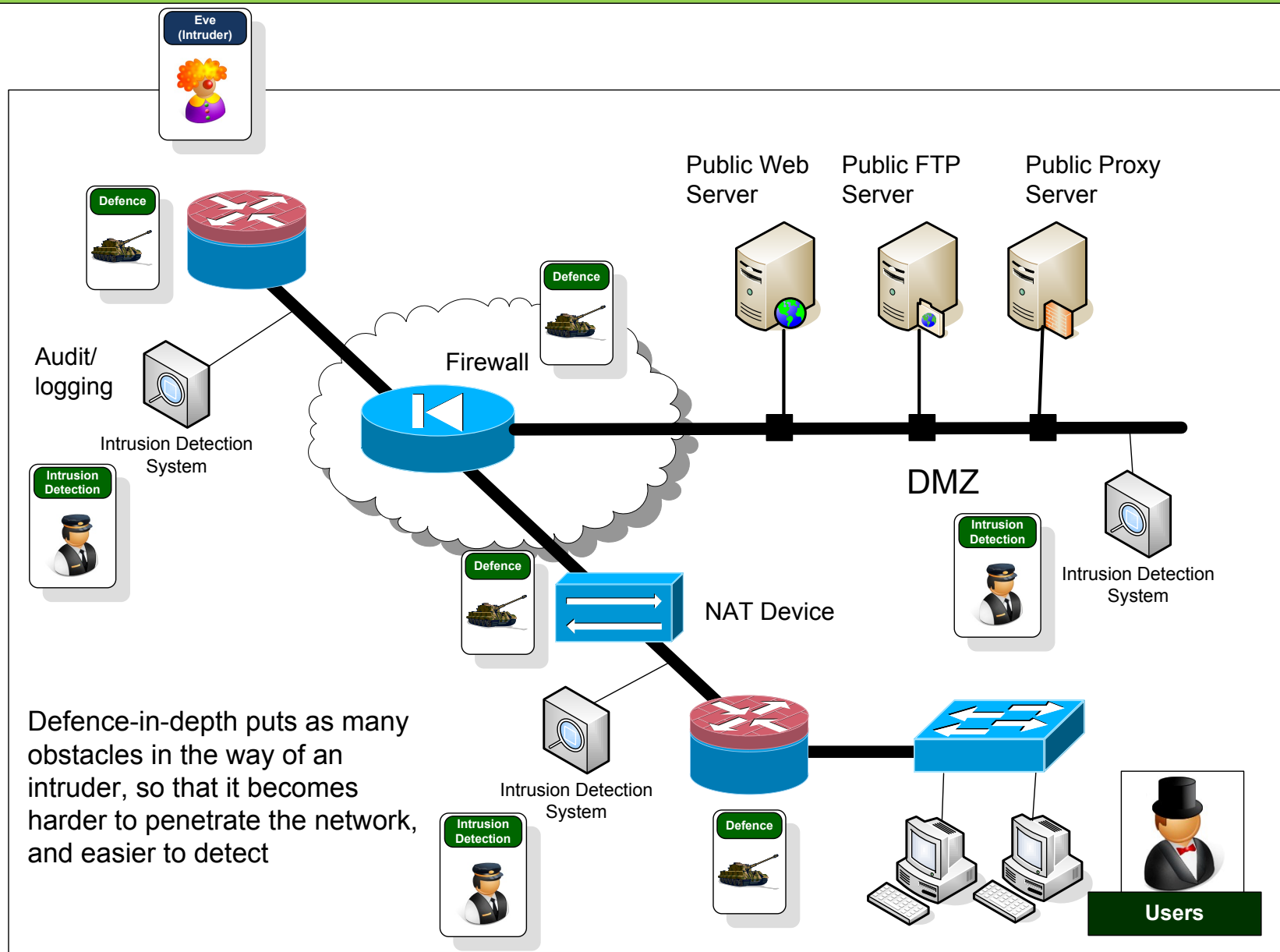
Terrorism/  
extortion

Fraud

Network/  
Organisational  
perimeter

**Internal threats (often a great threat than from outside)**





# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

Honeypots

IPS

Conclusions



Types



### Misuse Detection

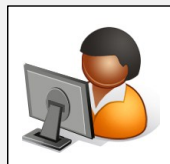
This attempts to model attacks on a system as specific patterns, and then scans for occurrences of these. Its disadvantage is that it struggles to detect new attacks.



Viruses/Worms



Denial-of-Service



External hack  
(scripting)

Intrusion  
Detection



IDS agent

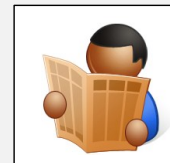


### Anomaly Detection.

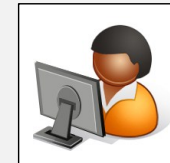
This assumes that abnormal behaviour by a user can be correlated with an intrusion. Its advantage is that it can typically react to new attacks, but can often struggle to detect variants of known attacks, particularly if they fit into the normal usage pattern of a user. Another problem is that the intruder can mimic the behavioural pattern of the user.



Personal abuse



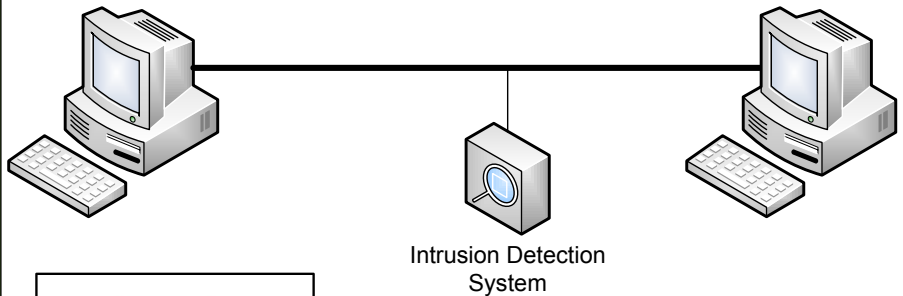
Fraud



External hack  
(human)



Data  
stealing



Network intrusion detection systems (NIDS)

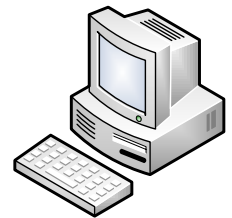
These **monitor packets** on the network and tries to determine an intrusion. This is either host base (where it runs on a host), or can listen to the network using a hub, router or probe.



System Integrity Verifier

These **monitor system files** to determine if an intruder has changed them (a backdoor attack). A good example of this is **Tripwire**. It can also watch other key system components, such as the Windows registry and root/administrator level privileges.

Intrusion Detection System



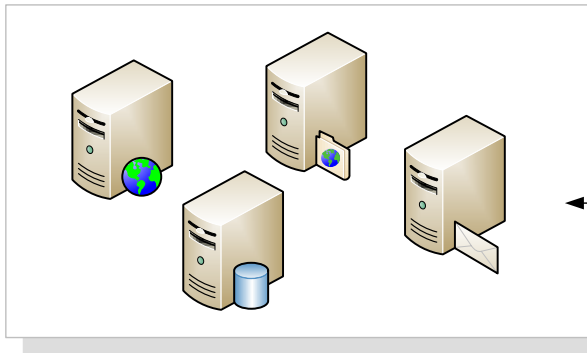
User profiling

29/05/2008	19:57	603	zhp1020.log
04/08/2007	12:00	337,920	zipfldr.dll
21/07/2008	03:00	102,400	ZLhp1020.dll
13/03/2008	15:46	53,248	zlib.dll
21/07/2008	03:00	28,672	zlm.dll

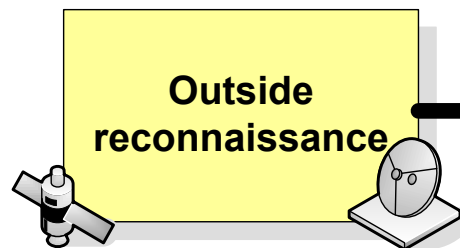


Log file monitors (LFM)

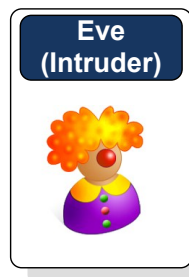
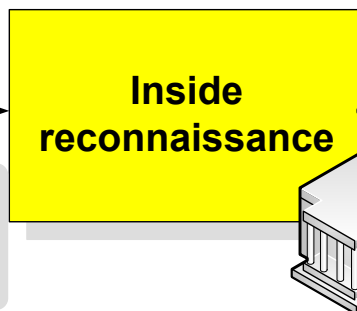
These monitor **log files** which are generated by network services, and look for key patterns of change. **Swatch** is a good example.



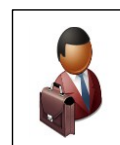
Intruder gains public information about the systems, such as DNS and IP information



Intruder gains more specific information such as subnet layout, and networked devices.



From code yellow to code red ...



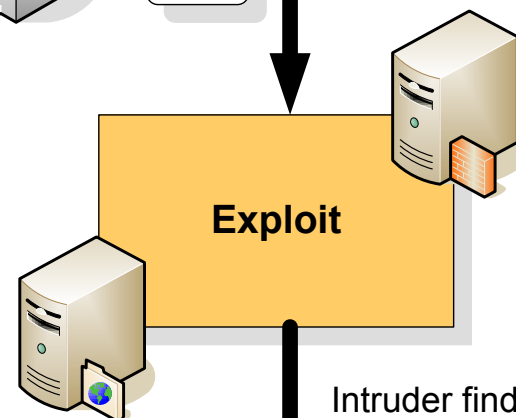
**Profit**

Data stealing, system damage, user abuse, and so on.



**Foothold**

Once into the system, the intruder can then advance up the privilege levels,



Intruder finds a weakness, such as cracking a password, breaching a firewall, and so on.



# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

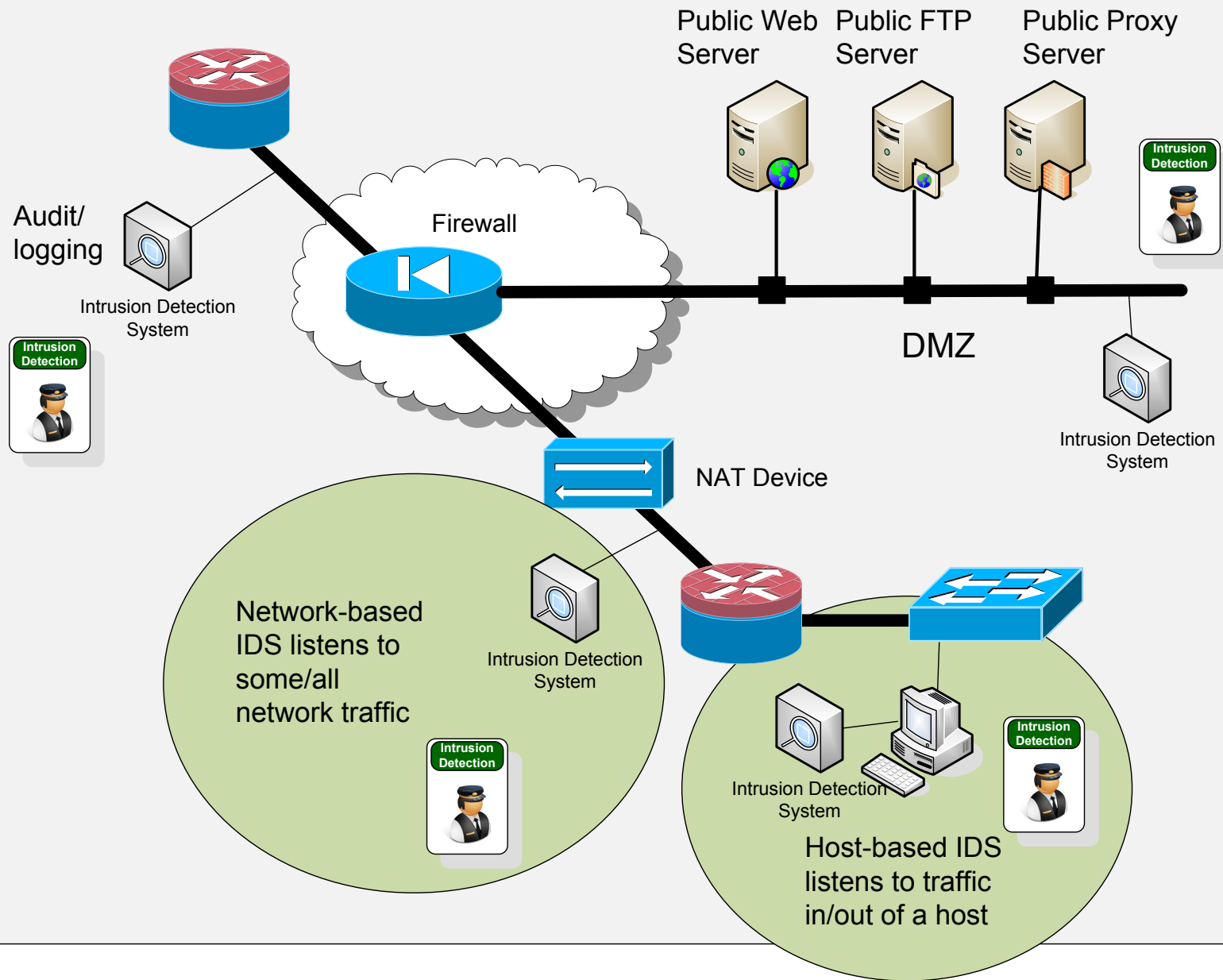
Honeypots

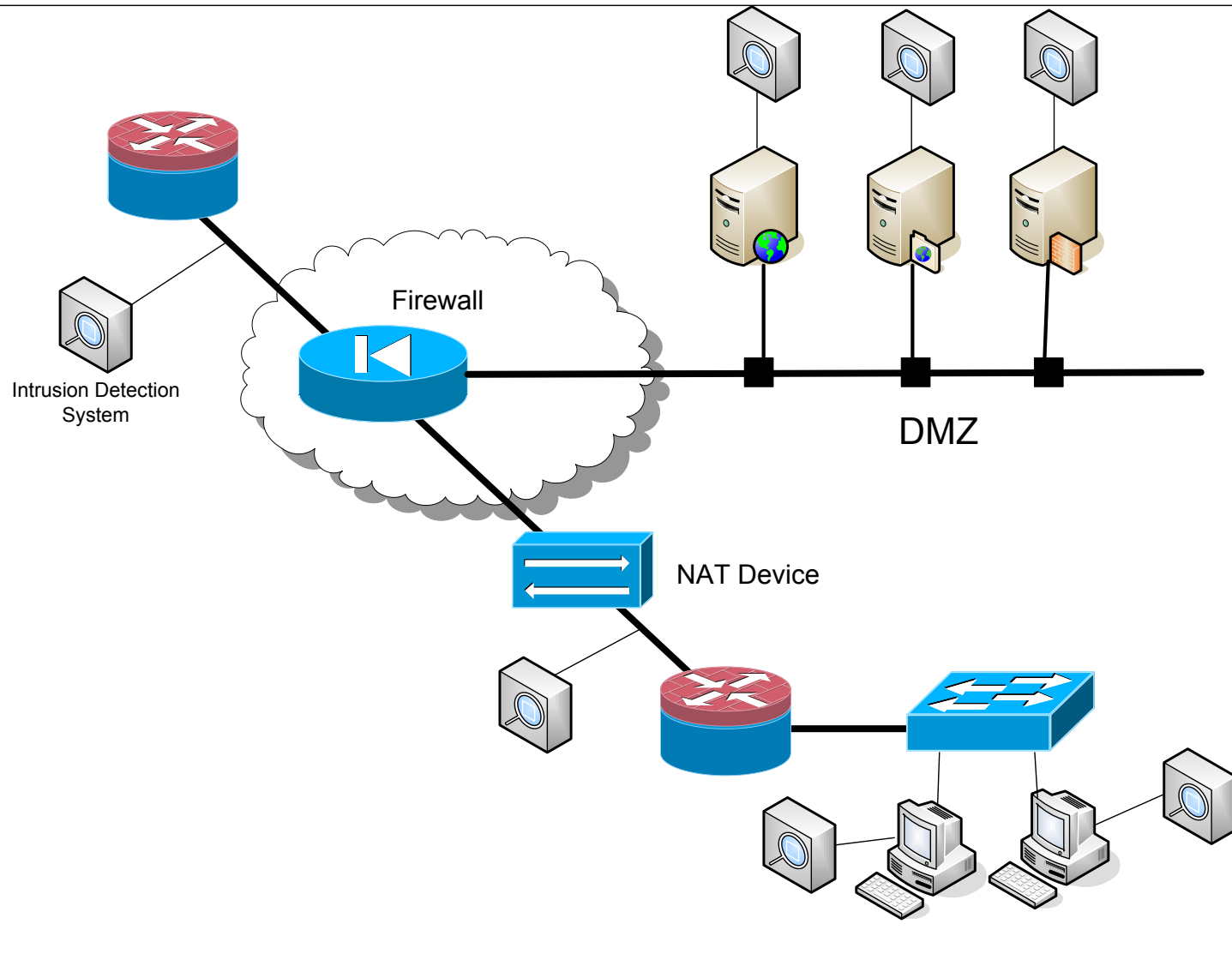
IPS

Conclusions

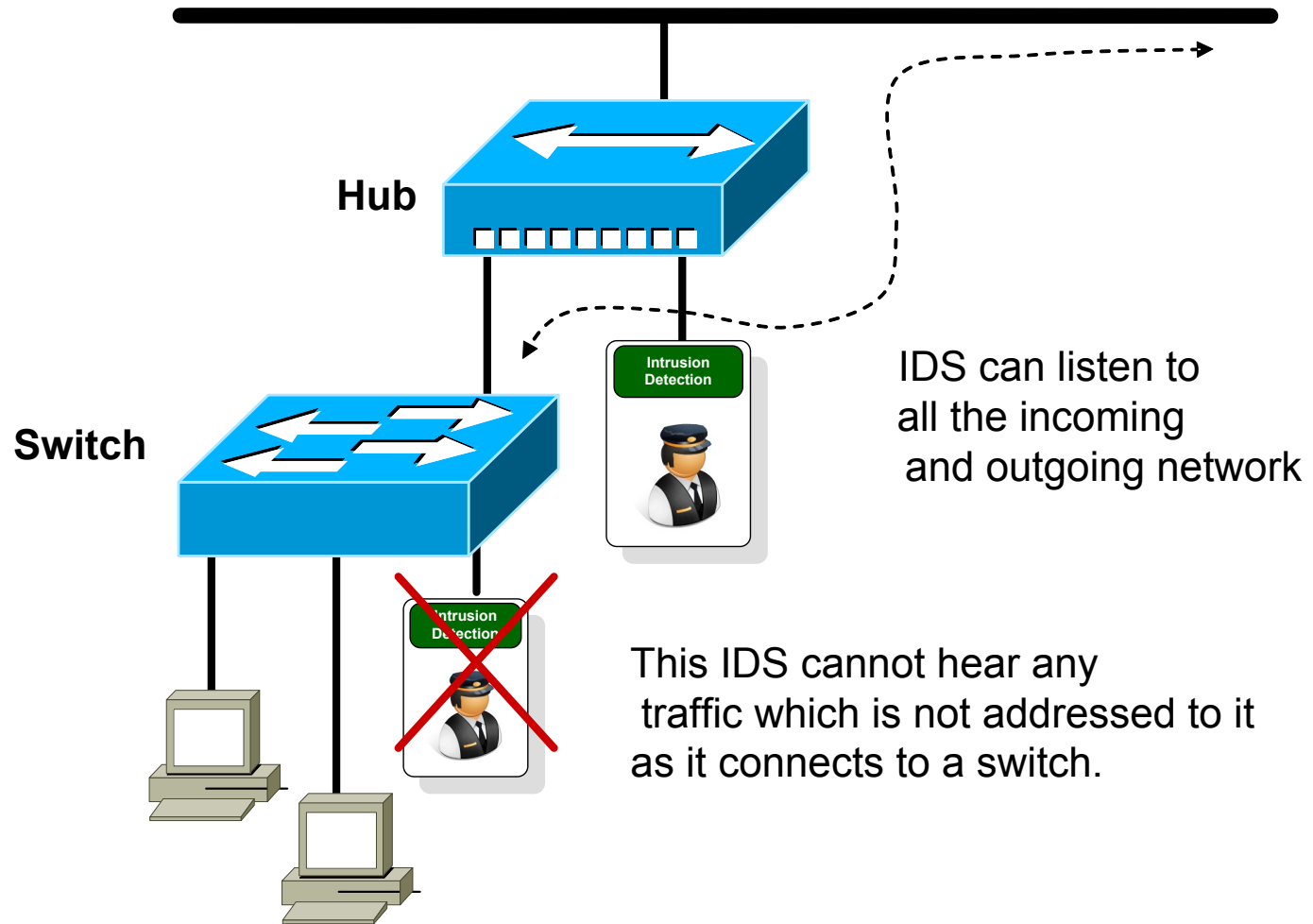


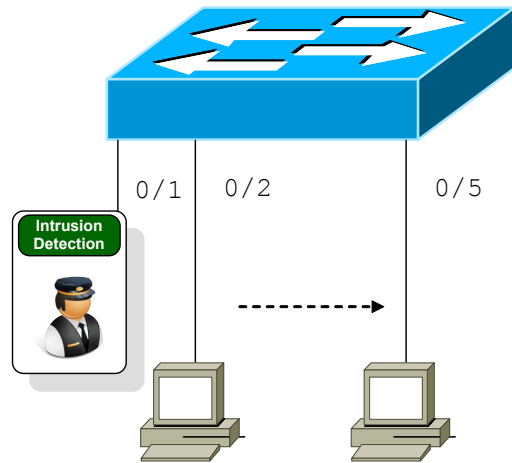
Host or Network?



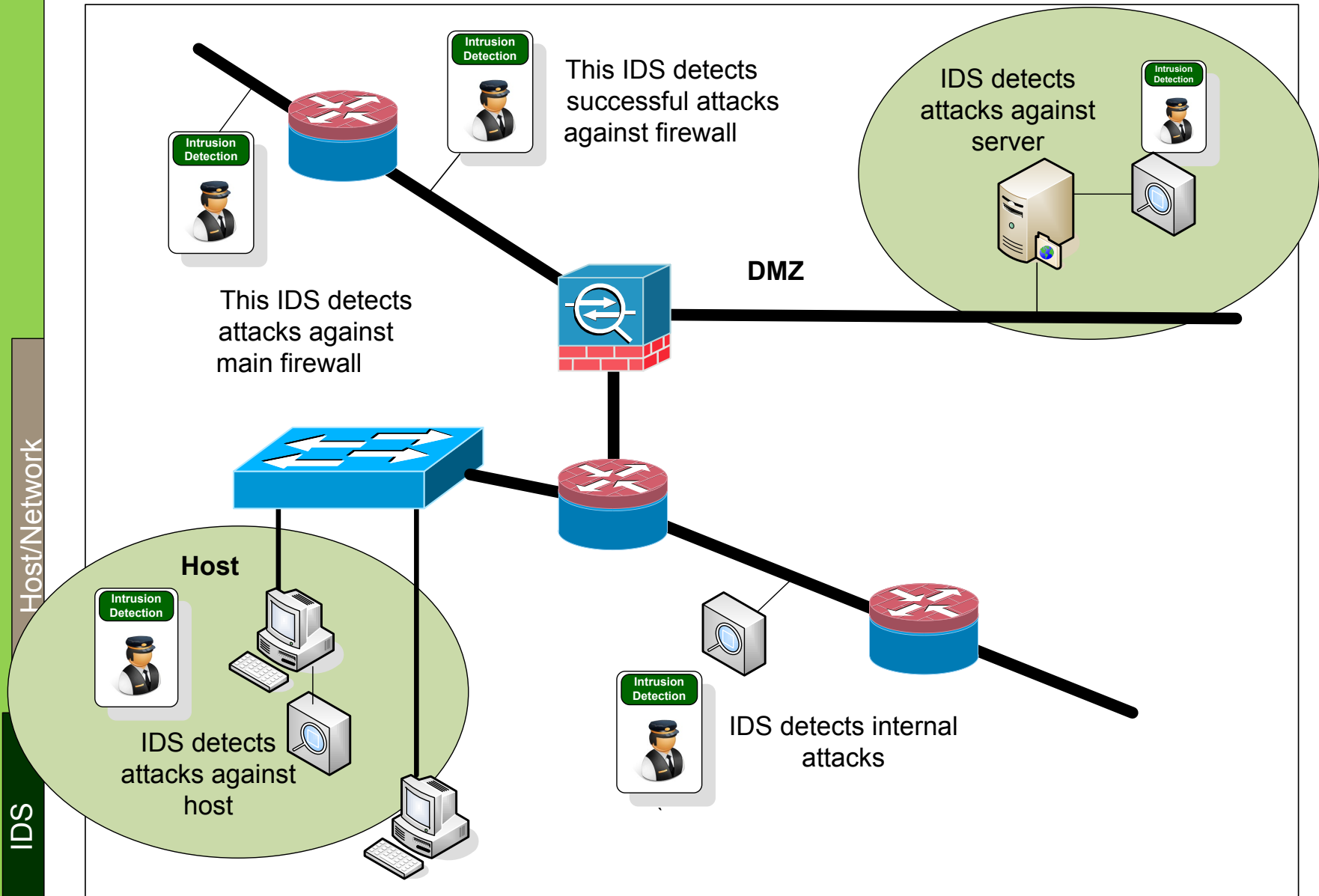








```
interface FastEthernet0/1
  port monitor FastEthernet0/2
  port monitor FastEthernet0/5
  port monitor VLAN2
!
interface FastEthernet0/2
!
interface FastEthernet0/3
  switchport access vlan 2
!
interface FastEthernet0/4
  switchport access vlan 2
!
interface FastEthernet0/5
!
interface VLAN1
  ip address 192.168.0.1 255.255.255.0
  no ip directed-broadcast
  no ip route-cache
!
```



# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

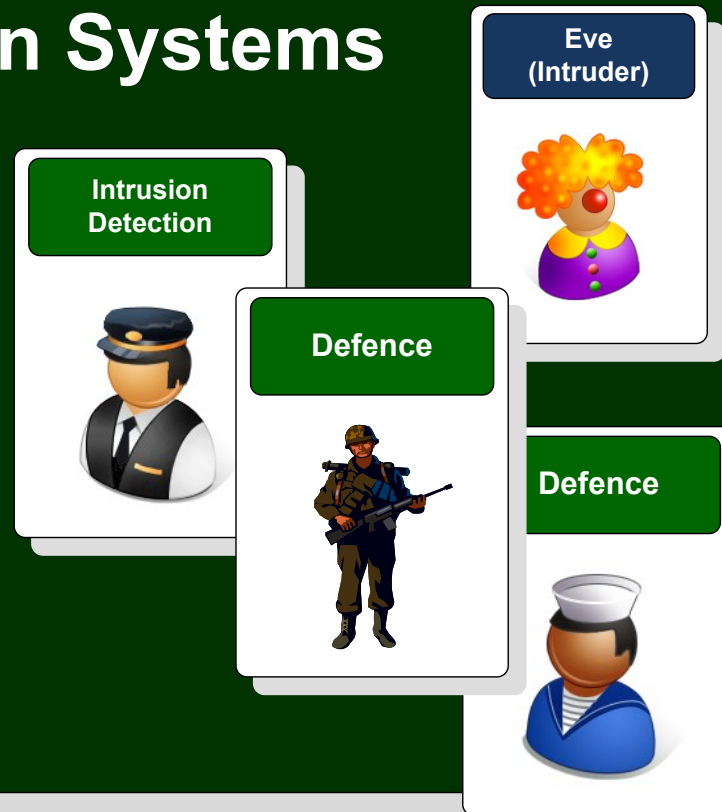
A few intrusions

User profiling

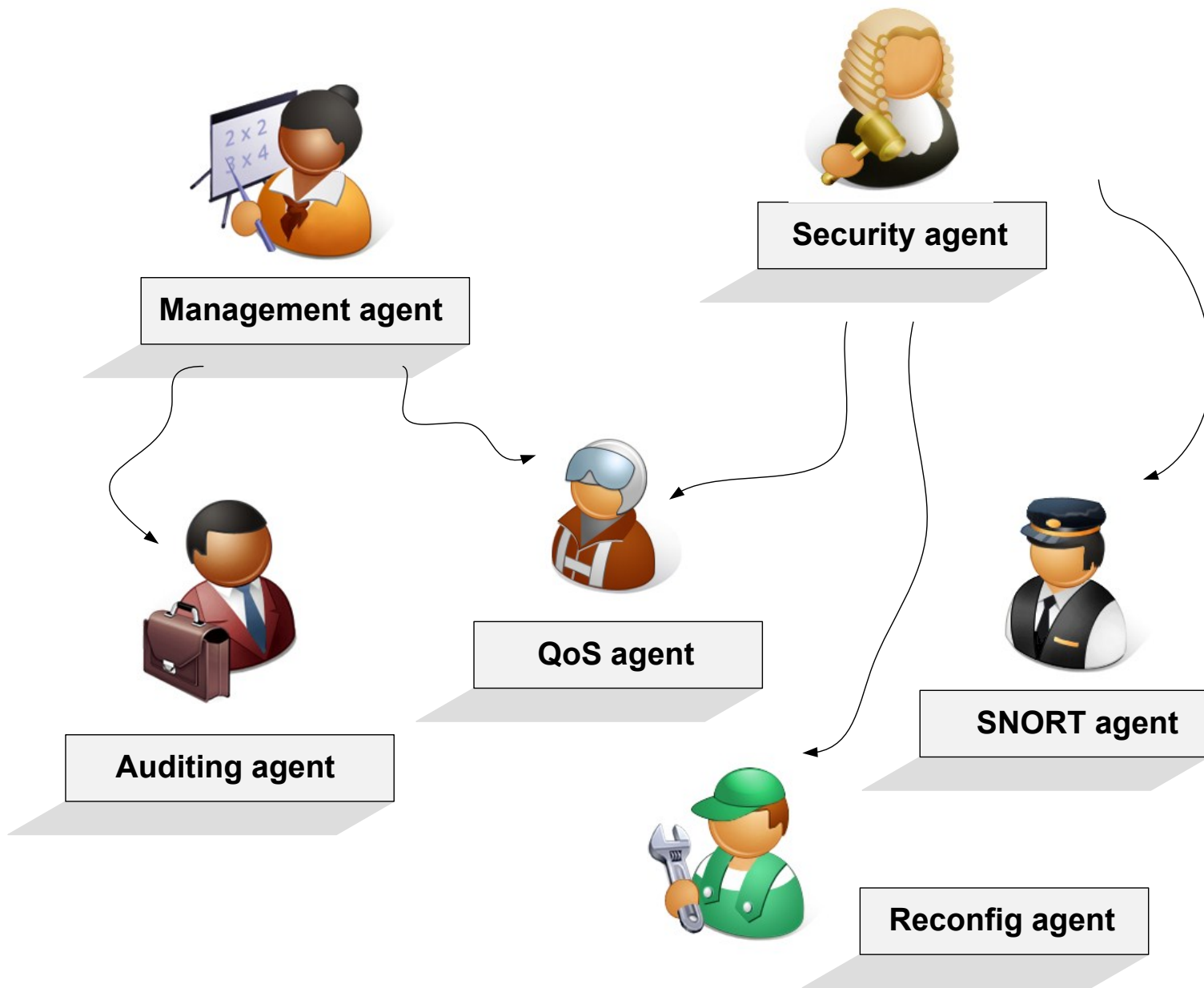
Honeypots

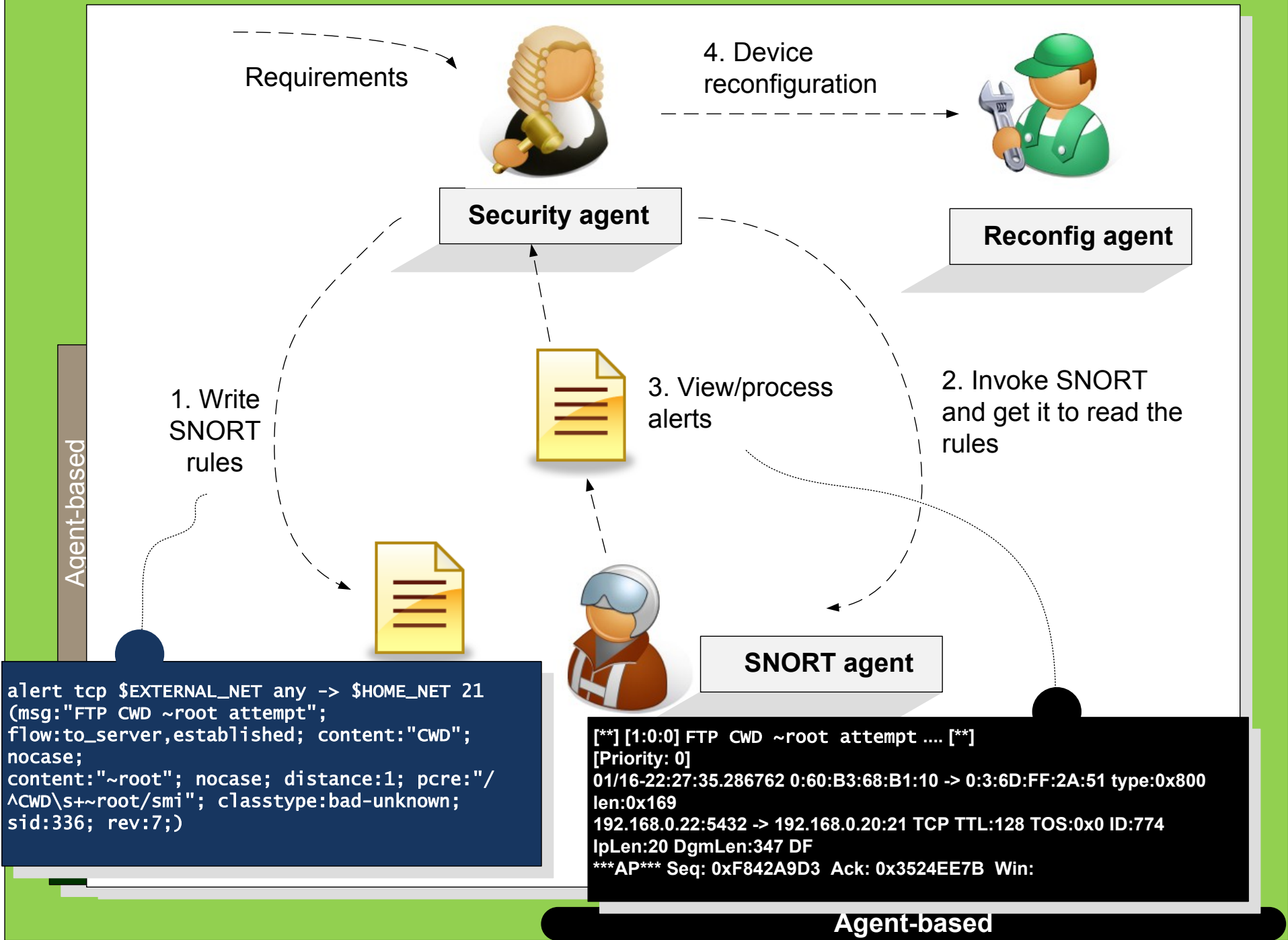
IPS

Conclusions



Agent-based





# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

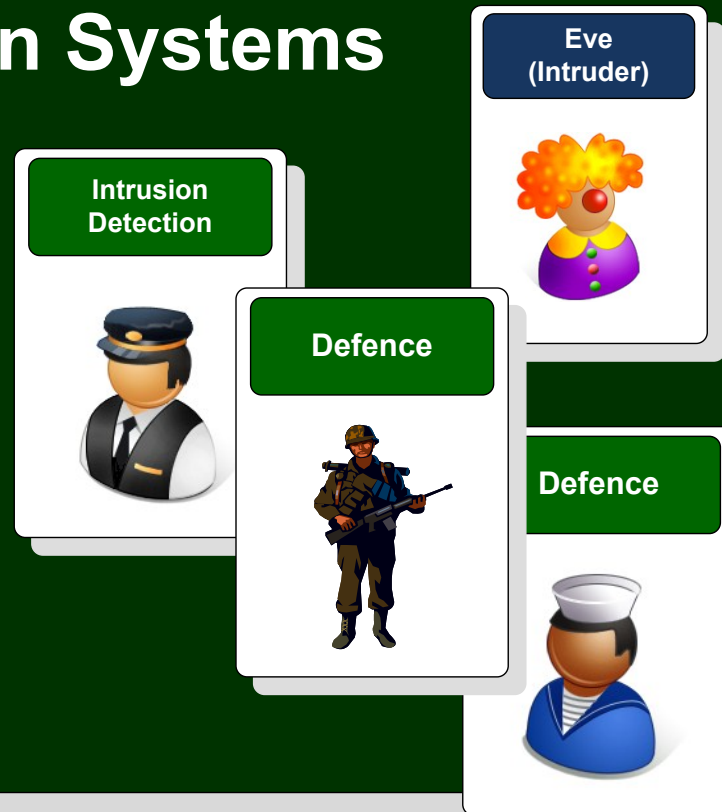
A few intrusions

User profiling

Honeypots

IPS

Conclusions



WinPCap

Wireshark



SNORT



User-defined  
agent



API Interface

Capture  
filter

WinPCap

WinPcap Pro



libpcap



Network Interface: Ethernet, Wireless, ADSL, etc

```
"ip"  
"icmp"  
"ip and tcp"  
"host 192.168.0.1"  
"host www.google.com"  
"tcp port 8080"  
"not tcp port http"  
"ether host 00:cc:dd:00:cc:dd"
```

Integrating with WinPCap – capturing packets



```
"ip"  
"icmp"  
"ip and tcp"  
"host 192.168.0.1"  
"host www.google.com"  
"tcp port 8080"  
"not tcp port http"  
"ether host 00:cc:dd:00:cc:dd"
```

Capture  
filter

User-defined  
agent



API  
Interface

Tamir Code Wrapper (.NET interface)

WinPcap

WinPcap Pro



Network Interface: Ethernet, Wireless, ADSL, etc

Integrating with WinPcap – capturing packets

WinPcap

IDS



```
using System;
using Tamir.IPLib;
namespace NapierCapture
{
    public class ShowDevices
```

```
    {
        public static void Main(string[] args)
        {
            string verWinPCap = null;
            int count = 0;
            verWinPCap = Tamir.IPLib.Version.GetVersionString();
            PcapDeviceList getNetConnections = SharpPcap.GetAllDevices();
            Console.WriteLine("WinPCap Version: {0}", verWinPCap);
            Console.WriteLine("Connected devices:\r\n");
            foreach (PcapDevice net in getNetConnections)
            {
                Console.WriteLine("{0}) {1}", count, net.PcapDescription);
                Console.WriteLine("\tName: \t{0}", net.PcapName);
                Console.WriteLine("\tMode: \t\t{0}", net.PcapMode);
                Console.WriteLine("\tIP Address: \t\t{0}", net.PcapIpAddress);
                Console.WriteLine("\tLoopback: \t\t{0}", net.PcapLoopback);
                Console.WriteLine();
                count++;
            }
            Console.WriteLine("Press any <RETURN> to exit");
            Console.ReadLine();
        }
    }
}
```

WinPCap

WinPCap Version: 1.0.2.0  
Connected devices:

- 0) Realtek RTL8169/8110 Family Gigabit Ethernet NIC  
(Microsoft's Packet Scheduler)  
Name: \Device\NPF\_{A22E93C1-A78D-4AFE-AD2B-517889CE42D7}  
Mode: Capture  
IP Address: 192.168.2.1  
Loopback: False
- 1) Intel(R) PRO/Wireless 2200BG Network Connection (Microsoft's Packet Scheduler)  
Name: \Device\NPF\_{044B069D-B90A-4597-B99E-A68C422D5FE3}  
Mode: Capture  
IP Address: 192.168.1.101  
Loopback: False

## Tamir Code Wrapper (.NET interface)

WinPcap Pro

WinPCap



rating with WinPCap – showing the interface

```

namespace NapierCapture
{
    public class CapturePackets
    {
        public static void Main(string[] args)
        {
            PcapDeviceList getNetConnections = SharpPcap.GetAllDevices();

            NetworkDevice netConn = (NetworkDevice)getNetConnections[1];
            PcapDevice device = netConn;
            device.PcapOnPacketArrival +=
                new SharpPcap.PacketArrivalEvent(device_PcapOnPacketArrival);
            Console.WriteLine("Network connection: {0}",
                device.PcapDescription);
            device.PcapStartCapture();
            Console.Write("Press any <RETURN> to exit");
            Console.Read();
            device.PcapStopCapture();
            device.PcapClose();
        }
        private static void device_PcapOnPacketArrival(object sender,
            Packet packet)
        {
            DateTime time = packet.PcapHeader.Date;
            int len = packet.PcapHeader.PacketLength;
            Console.WriteLine("{0}:{1}:{2},{3} Len={4}",time.Hour,
                time.Minute, time.Second, time.Millisecond, len);
        }
    }
}

```



## Tamir Code Wrapper (.NET interface)

**WinPcap**

WinPcap  Pro



```

13:17:56,990 Len=695
13:17:57,66 Len=288
13:17:57,68 Len=694
13:18:4,363 Len=319
13:18:4,364 Len=373
13:18:4,364 Len=371
13:18:4,365 Len=375
13:18:4,366 Len=367

```

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

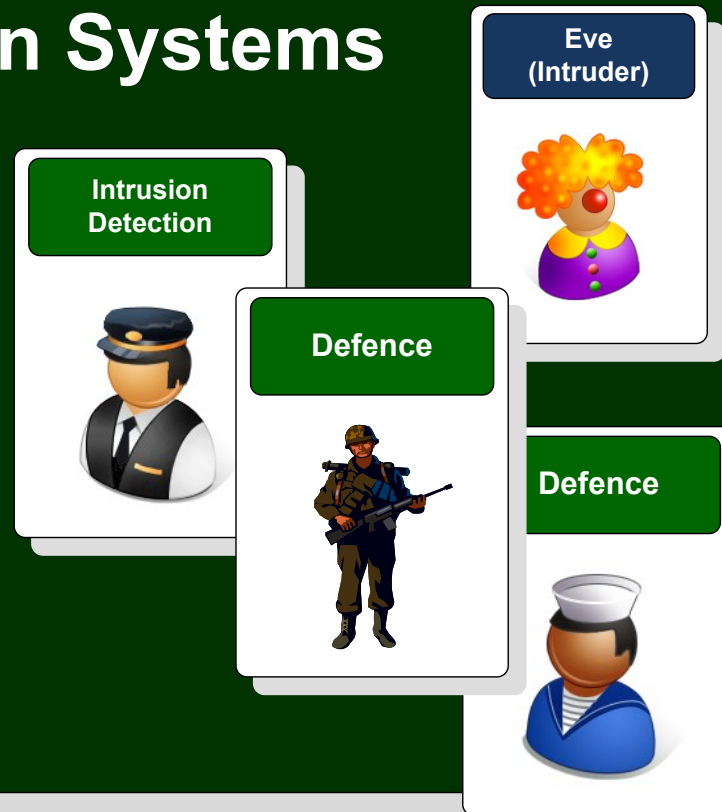
A few intrusions

User profiling

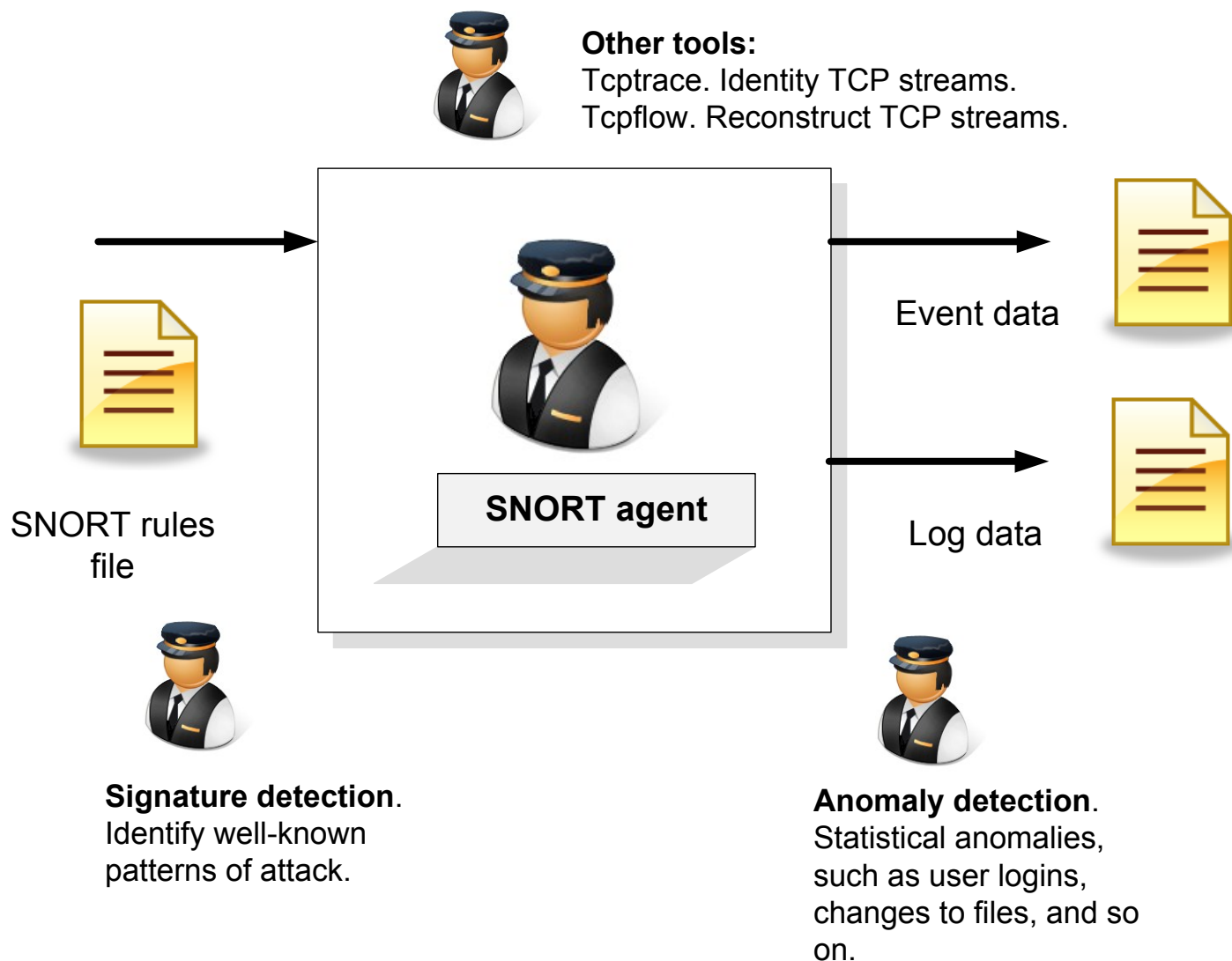
Honeypots

IPS

Conclusions

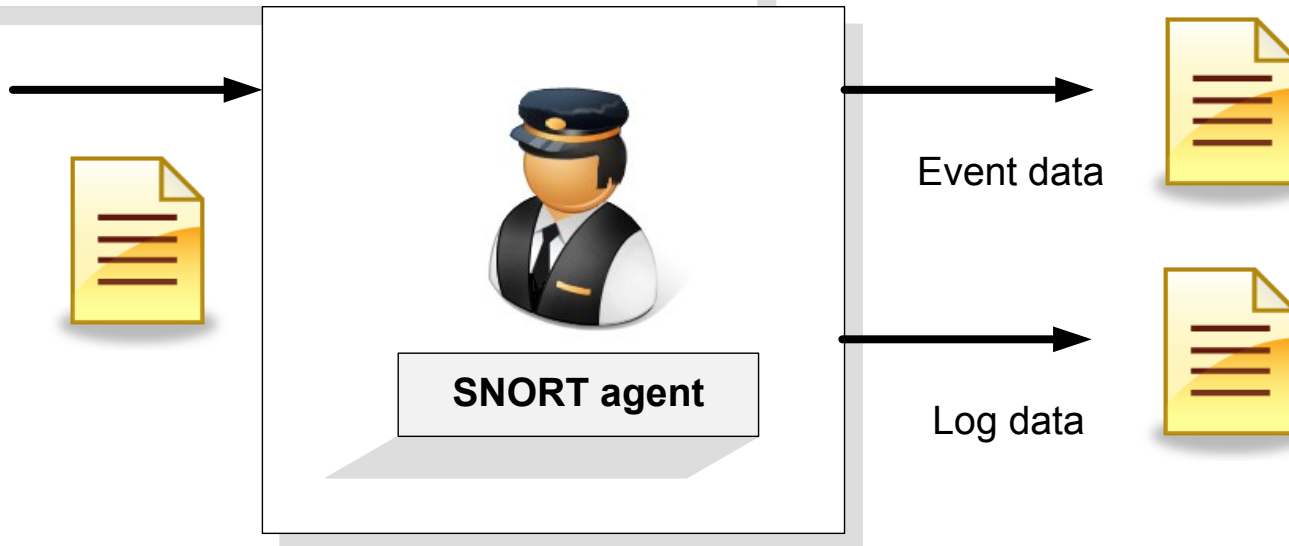


Snort



```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"mountd access");
```

alert	Generate an alert and log packet
log	Log packet
pass	Ignore the packet
activate	Alert and activate another rule
Dynamic	Remain idle until activated by an activate rule



Snort

IDS

Snort rules

```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"mountd access";)
```

[Source IP] [Port]

[Destination IP] [Port]



**SNORT agent**



Event data



Log data



Snort

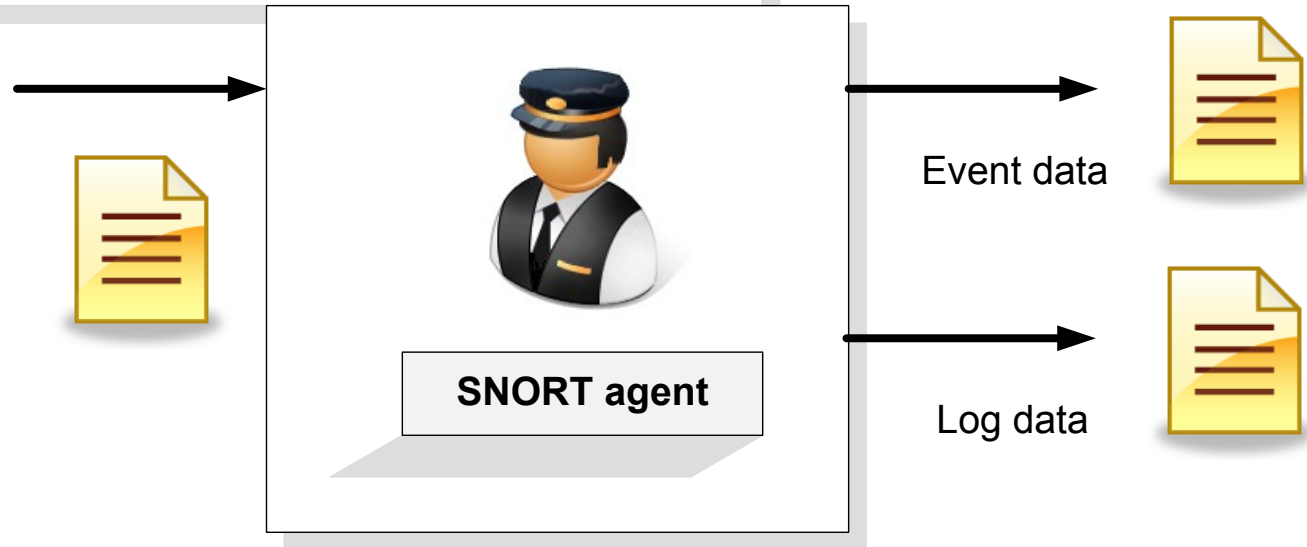
IDS

**Snort rules**

```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"moundd access");
```

**Payload detection:**  
**Hex sequence**    "|00 01 86 a5|"  
**Text sequence**   "USER root"

**Modifiers:**  
rawbytes  
offset  
distance  
within  
uricontent  
bytejump





```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg:"mountd access";)
```



Message-to-display



SNORT agent

Event data



Log data

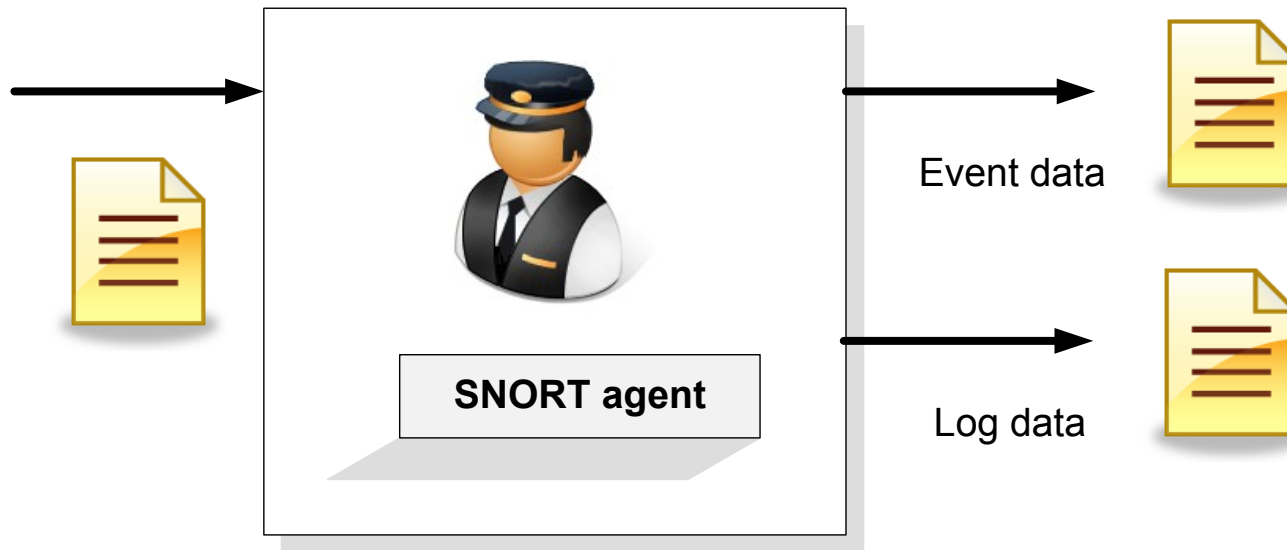


Snort

IDS

Payload detection

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 1863
(msg:"CHAT MSN login attempt"; flow:to_server,established; content:"USR "; depth:4;
nocase; content:" TWN "; distance:1; nocase;
classtype:policy-violation; sid:1991; rev:1;)
```



The SID and REV represent known Snort rules:

- Less 100 Reserved for future use
- Between 100 and 1,000,000 are rules included with the Snort distribution
- More than 1,000,000 is for local rules

For example: **sid:336; rev:7;** represents an attempt to change to the system administrator's account in FTP.

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

Honeypots

IPS

Conclusions



A simple rule

```
alert tcp any any -> any any (content:"the"; msg:"The found ....");
```

```
Snort -v -c bill.rules -l /log
```

Alert.ids  
(in \log)

**16 January 10:27pm**

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

01/16-22:27:35.286762 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x169

192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:774 IpLen:20

DgmLen:347 DF

\*\*\*AP\*\*\* Seq: 0xF842A9D3 Ack: 0x3524EE7B Win: 0x4321 TcpLen: 20

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

01/16-22:27:35.287084 0:3:6D:FF:2A:51 -> 0:60:B3:68:B1:10 type:0x800 len:0x198

192.168.0.20:3554 -> 192.168.0.22:445 TCP TTL:128 TOS:0x0 ID:1086 IpLen:20

DgmLen:394 DF

\*\*\*AP\*\*\* Seq: 0x3524EE7B Ack: 0xF842AB06 Win: 0x42E4 TcpLen: 20

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

01/16-22:27:35.290026 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x5D

192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:775 IpLen:20

DgmLen:79 DF

\*\*\*AP\*\*\* Seq: 0xF842AB06 Ack: 0x3524EFDD Win: 0x41BF TcpLen: 20

## IP header

Version	Header len.	Type of service
Total length		
Identification		
0	D	M
Fragment Offset		
Time-to-live (TTL)		Protocol
Header Checksum		
Source IP Address		
Destination IP Address		

## TCP header

TCP Source Port	
TCP Destination Port	
Sequence Number	
Acknowledgement Number	
Data Offset	Flags/Reserved
Window	
Checksum	
Urgent Pointer	

## Ethernet frame

	Src MAC address	Dest. MAC address	Type	Length	IP header	TCP header	Data	
--	-----------------	-------------------	------	--------	-----------	------------	------	--

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

**01/16-22:27:35.286762 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x169**

192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:774 IpLen:20 DgmLen:347 DF

\*\*\*AP\*\*\* Seq: 0xF842A9D3 Ack: 0x3524EE7B Win: 0x4321 TcpLen: 20

A simple rule

IDS

Payload detection

## IP header

Version	Header len.	Type of service
Total length		
Identification		
0	D	M
Fragment Offset		
Time-to-live (TTL)		Protocol
Header Checksum		
Source IP Address		
Destination IP Address		

## TCP header

TCP Source Port	
TCP Destination Port	
Sequence Number	
Acknowledgement Number	
Data Offset	Flags/Reserved
Window	
Checksum	
Urgent Pointer	

## Ethernet frame

Src MAC address	Dest. MAC address	Type	Length	IP header	TCP header	Data	
-----------------	-------------------	------	--------	-----------	------------	------	--

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

01/16-22:27:35.286762 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x169

**192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:774 IpLen:20 DgmLen:347 DF**

\*\*\*AP\*\*\* Seq: 0xF842A9D3 Ack: 0x3524EE7B Win: 0x4321 TcpLen: 20

### Protocol

1 ICMP  
6 TCP  
8 EGP  
41 IPv6 over IPv4  
46 RSVP

51 AH

Payload detection

A simple rule

IDS

## IP header

Version	Header len.	Type of service
Total length		
Identification		
0	D	M
Fragment Offset		
Time-to-live (TTL)		Protocol
Header Checksum		
Source IP Address		
Destination IP Address		

## TCP header

TCP Source Port	
TCP Destination Port	
Sequence Number	
Acknowledgement Number	
Data Offset	Flags/Reserved
Window	
Checksum	
Urgent Pointer	

## Ethernet frame

Src MAC address	Dest. MAC address	Type	Length	IP header	TCP header	Data	
-----------------	-------------------	------	--------	-----------	------------	------	--

[\*\*] [1:0:0] The found .... [\*\*]

[Priority: 0]

01/16-22:27:35.286762 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x169

192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:774 IpLen:20 DgmLen:347 DF

\*\*\*AP\*\*\* Seq: 0xF842A9D3 Ack: 0x3524EE7B Win: 0x4321 TcpLen: 20



**TCP\_3423-445.ids**

**TCP\_3424-139.ids**

**TCP\_3521-445.ids**

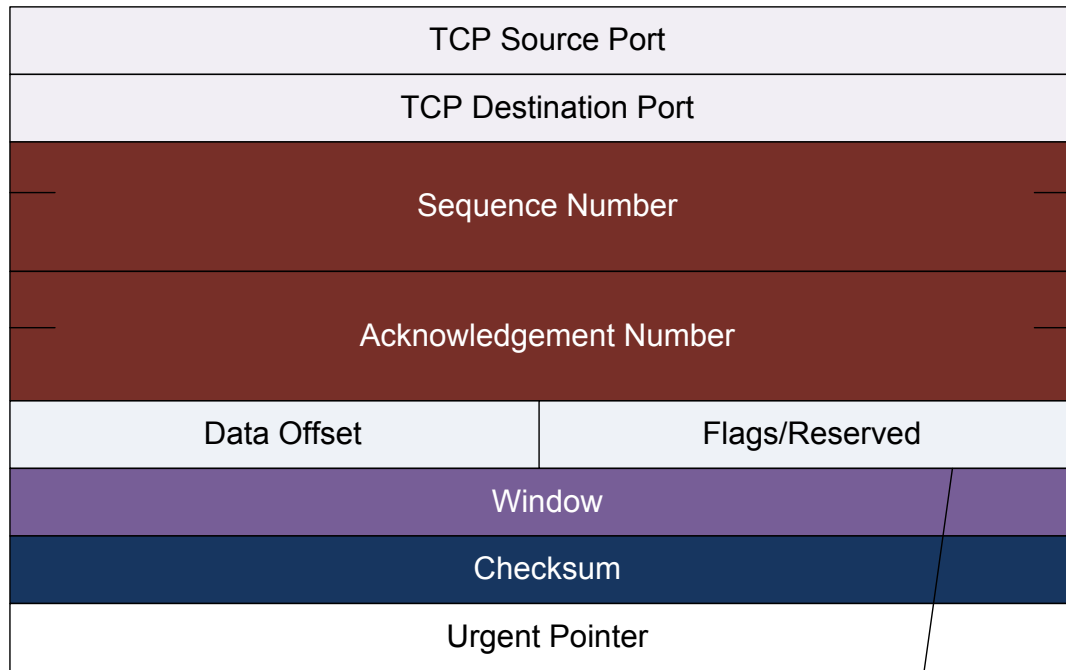
**TCP\_3529-139.ids**

**TCP\_3554-445.ids**

# TCP\_3566-445.ids

Log of traffic  
between port 3423 and  
455 to/from 192.168.0.20[illegible]



**TCP header**

Flags – the flag field is defined as UAPRSF,

- U is the urgent flag (URG).
- A the acknowledgement flag (ACK).
- P the push function (PSH).
- R the reset flag (RST).
- S the sequence synchronize flag (SYN).
- F the end-of-transmission flag (FIN).

Originator

1. CLOSED

2. SYN-SENT -> <SEQ=999><CTL=SYN>

3. ESTABLISHED <SEQ=100><ACK=1000><CTL=SYN,ACK> <-

4. ESTABLISHED -> <SEQ=1000><ACK=101> <CTL=ACK>

5. ESTABLISHED -> <SEQ=1000><ACK=101> <CTL=ACK><DATA>

Recipient

LISTEN

SYN-RECEIVED

SYN-RECEIVED

ESTABLISHED

ESTABLISHED

The SYN flag identifies  
a connection

Flags – the flag field is defined as UAPRSF,

- U is the urgent flag (URG).
- A the acknowledgement flag (ACK).
- P the push function (PSH).
- R the reset flag (RST).
- S the sequence synchronize flag (SYN).
- F the end-of-transmission flag (FIN).

An incoming SYN flag is important in detecting the start of a connection. The main flags are:

**F** FIN  
**S** SYN  
**R** RST  
**P** PSH  
**A** ACK  
**U** URG

The following modifiers can be set to change the match criteria:

- + match on the specified bits, plus any others
- \* match if any of the specified bits are set
- ! match if the specified bits are not set

Example to test for SYN flag:

```
alert tcp any any -> any any (flags:S;)
```

It is often important to know the flow direction. The main flow rules options are:

- to\_client. Used for server responses to client.
- to\_server. Used for client requests to server.
- from\_client. Used on client responses.
- from\_server. Used on server responses.
- established. Established TCP connections.

Example to test for an FTP connection to the users computer:

```
alert tcp any any -> $HOME_NET 21 (flow: from_client;  
content: "CWD incoming"; nocase;
```

IP header

Version			Header len.		Type of service	
Total length						
Identification						
0	D	M	Fragment Offset			
Time-to-live (TTL)				Protocol		
Header Checksum						
Source IP Address						
Destination IP Address						

TCP header

TCP Source Port	
TCP Destination Port	
Sequence Number	
Acknowledgement Number	
Data Offset	Flags/Reserved
Window	
Checksum	
Urgent Pointer	

01/16-22:11:15.833440 192.168.0.20:3423 -> 192.168.0.22:445

TCP TTL:128 TOS:0x0 ID:975 IpLen:20 DgmLen:48 DF

\*\*\*\*\***S**\* Seq: 0x26885B8B Ack: 0x0 Win: 0x4000 TcpLen: 28

TCP Options (4) => MSS: 1460 NOP NOP SackOK

+++++

01/16-22:11:15.835497 192.168.0.22:445 -> 192.168.0.20:3423

TCP TTL:128 TOS:0x0 ID:653 IpLen:20 DgmLen:48 DF

\*\*\***A\*\*S**\* Seq: 0xE9A4004C Ack: 0x26885B8C Win: 0x4470 TcpLen: 28

TCP Options (4) => MSS: 1460 NOP NOP SackOK

+++++

01/16-22:11:15.835571 192.168.0.20:3423 -> 192.168.0.22:445

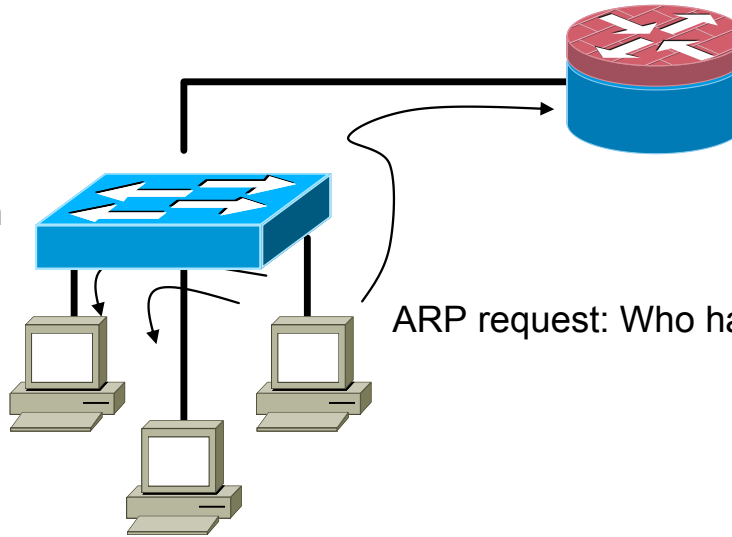
TCP TTL:128 TOS:0x0 ID:977 IpLen:20 DgmLen:40 DF

\*\*\***A\*\*\*\***\* Seq: 0x26885B8C Ack: 0xE9A4004D Win: 0x4470 TcpLen: 20

+++++

Devices can only communicate directly if they have the MAC address and IP address.

Switch



ARP request: Who has 192.168.0.168?



ARP request is broadcast to the network

```
01/16-09:31:08.785149 ARP who-has 192.168.0.168 tell 192.168.0.22
01/16-09:45:59.458607 ARP who-has 192.168.0.42 tell 192.168.0.216
01/16-09:45:59.459159 ARP reply 192.168.0.42 is-at 0:20:18:38:B8:63
01/16-09:46:03.857325 ARP who-has 192.168.0.104 tell 192.168.0.198
01/16-09:46:10.125715 ARP who-has 192.168.0.15 tell 192.168.0.38
01/16-09:46:10.125930 ARP who-has 192.168.0.38 tell 192.168.0.15
```

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

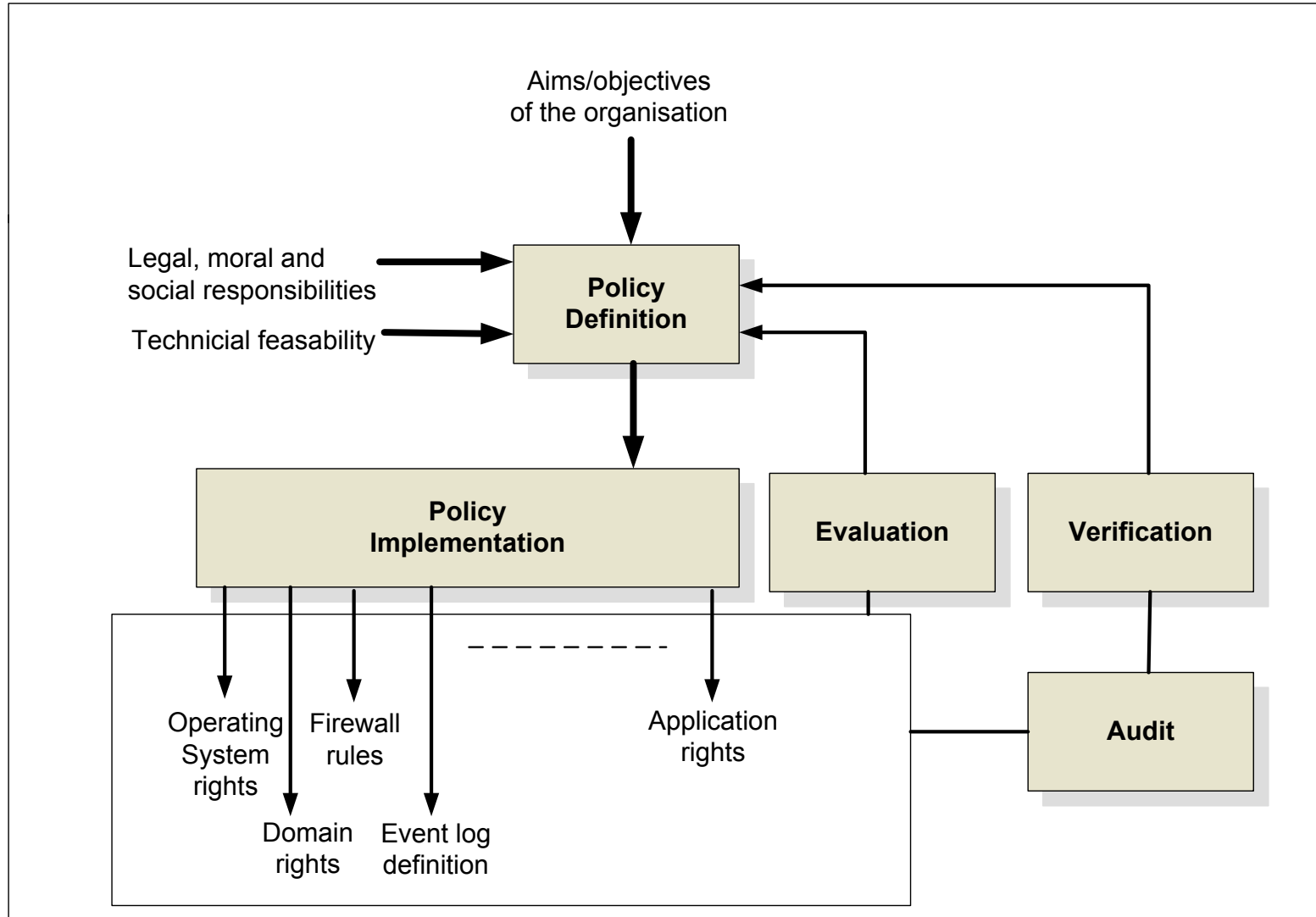
Honeypots

IPS

Conclusions



A few intrusions



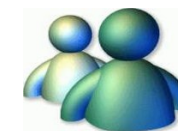
```

alert tcp $HOME_NET any -> $EXTERNAL_NET 1863
(msg:"CHAT MSN login attempt"; flow:to_server,established; content:"USR "; depth:4;
nocase; content:" TWN "; distance:1; nocase;
classtype:policy-violation; sid:1991; rev:1;)

```



No.	Source	Destination	Protocol	Info
1	192.168.0.3	207.46.28.93	TCP	5398 > 1863 [SYN] Seq=0 Len=0 MSS=1460
2	207.46.28.93	192.168.0.3	TCP	1863 > 5398 [SYN, ACK] Seq=0 Ack=1 win=5840
3	192.168.0.3	207.46.28.93	TCP	5398 > 1863 [ACK] Seq=1 Ack=1 win=17520
4	192.168.0.3	207.46.111.39	MSNMS	USR 2 TWN I test@hotmail.com
5	207.46.111.39	192.168.0.3	MSNMS	USR 26 OK test@hotmail.com 1 0



```

private static void device_PcapOnPacketArrival(...)
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)packet;
        int destPort = tcp.SourcePort; byte [] b = tcp.Data;
        ASCIIEncoding format = new ASCIIEncoding();
        string s = format.GetString(b); s=s.ToLower();
        if (destPort==1863 && (s.StartsWith("usr ") && s.IndexOf(" twn ")>0 )
            Console.WriteLine("MSN Messenger Login");
    }
}

```





```
alert tcp $HOME_NET any -> $EXTERNAL_NET 1863
```

```
(msg:"CHAT MSN login attempt";
```

```
flow:to_server,established;
```

```
content: "USR "; depth:4; nocase;
```

```
content:" TWN "; distance:1; nocase;
```

```
classtype:policy-violation;sid:1991; rev:1;)
```

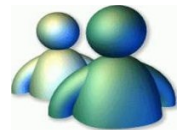
DEPTH. Modifier for previous content ... defines to look within 4 bytes of the payload

NOCASE; Modifier for previous content ... ignore the case of the content

DISTANCE. Modified for previous content and defines how far into the payload it should search (in bytes)

No.	Source	Destination	Protocol
1	192.168.0.3	207.46.28.93	TCP
2	192.168.0.3	207.46.28.93	TCP
3	207.46.28.93	192.168.0.3	TCP
4	192.168.0.3	207.46.28.93	TCP
5	192.168.0.3	207.46.111.39	MSNMS
6	207.46.111.39	192.168.0.3	MSNMS

```
1863 > 5398 [SYN, ACK] Seq=0 Ack=1 Win=5840
5398 > 1863 [ACK] Seq=1 Ack=1 Win=17520
USR 2 TWN I test@hotmail.com
USR 26 OK test@hotmail.com 1 0
```



```
alert tcp $HOME_NET any -> $EXTERNAL_NET
```

```
(msg:"CHAT MSN login attempt";
```

```
flow:to_server,established;
```

```
content: "USR "; depth:4; nocase;
```

```
content:" TWN "; distance:1; nocase;
```

```
classtype:policy-violation; sid:1991
```

```
icmp-event
misc-activity
network-scan
not-suspicious
protocol-command-decode
string-detect
```

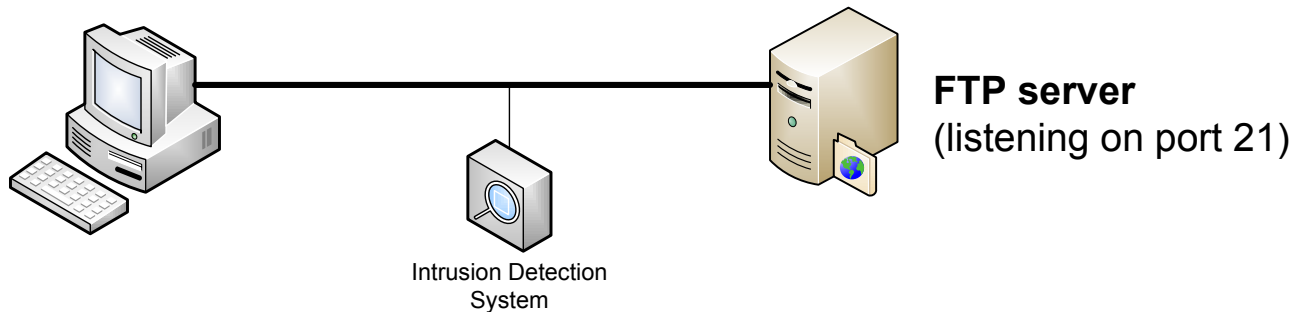
```
attempted-admin
attempted-user
policy-violation
shellcode-detect
successful-admin
successful-user
trojan-activity
unsuccessful-user
web-application-attack
```

```
attempted-dos
attempted-recon
bad-unknown
default-login-attempt
denial-of-service
misc-attack
non-standard-protocol
rpc-portmap-decode
successful-dos
successful-recon-largescale
...
web-application-activity
```

```
No. 1
1 1
2 1
3 2
4 1
5 1
6 2
```

```
> 1863 [S
> 1863 [S
> 5398 [S
> 1863 [A
2 TWN I te
26 OK test
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21
(msg:"FTP CWD ~root attempt"; flow:to_server,established; content:"CWD";
nocase; content:"~root"; nocase; distance:1; pcre:"/^CWD\s+~root/smi";
classtype:bad-unknown; sid:336; rev:7;)
```



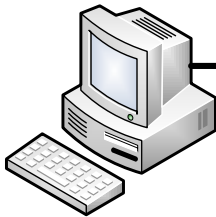
```
private static void device_PcapOnPacketArrival(..)
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)packet;
        int destPort = tcp.SourcePort;
        byte [] b = tcp.Data;
        ASCIIEncoding format = new ASCIIEncoding();
        string s = format.GetString(b); s=s.ToLower();
        if (destPort==21 && s.IndexOf("cwd")>0 && s.IndexOf("~root")>0 )
            Console.WriteLine("FTP CWD ~root attempt");
    }
}
```



```

alert tcp $EXTERNAL_NET any
(msg:"FTP CWD ~root attempt
nocase; content:"~root"; no
classtype:bad-unknown; sid:

```



```

private static void device_Pcap
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)
        int destPort = tcp.SourcePort;
        byte [] b = tcp.Data;
        ASCIIEncoding format = new ASCIIEncoding();
        string s = format.GetString(b);
        if (destPort==21 && s.IndexOf("CWD ~root") != -1)
            Console.WriteLine("FTP CWD ~root attempt");
    }
}

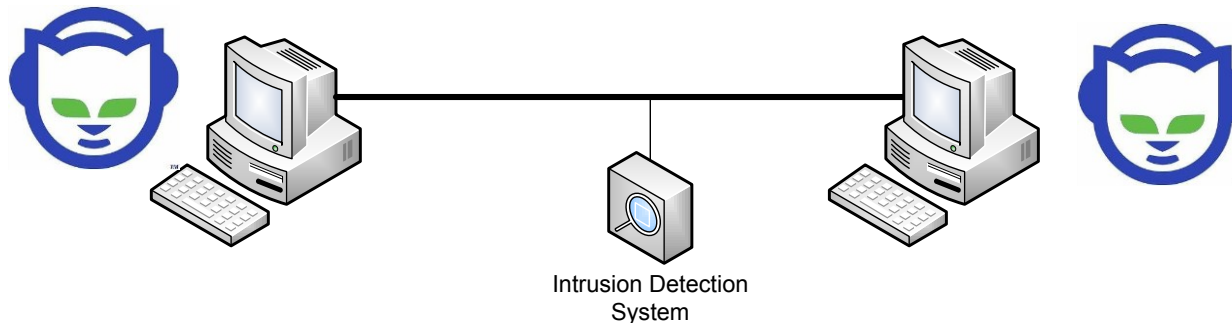
```

```

> telnet ftp.test.com 21
220-Microsoft FTP Service
220 NTXPW35
c wd
530 Please login with USER and PASS.
user bill
331 Password required for *.
pass *****
230-FTP Server
230 User bill logged in.
help
214-The following commands are recognized
ABOR      ACCT      ALLO      APPE      CDUP
CWD        DELE      FEAT      HELP
LIST
MDTM      MKD       MODE      NLST      NOOP
OPTS      PASS      PASV      PORT      PWD
QUIT      REIN      REST      RETR
RMD       RNFR      RNT0      SITE
SIZE      SMNT      STAT      STOR
STOU      STRU      SYST      TYPE
USER      XCUP      XCWD      XMKD
XPWD      XRMD
214  HELP command successful.
pwd
257 "/bill" is current directory.
c wd /
250 CWD command successful.
pwd
257 "/" is current directory.
c wd ~root
250 CWD command successful.
list
150 Opening ASCII mode data connection for /
bin/ls.

```

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 8888
(msg:"P2P napster login"; flow:to_server,established;
content:"|00 0200|"; offset:1; depth:3;
classtype:policy-violation; sid:549; rev:6;)
```



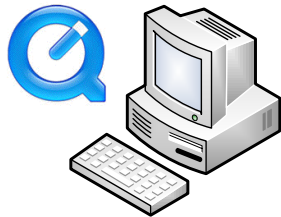
```
private static void device_PcapOnPacketArrival(...)
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)packet;
        int destPort = tcp.SourcePort;
        byte [] b = tcp.Data;
        if (destPort==8888 && b[0]==0x00 && b[1]==0x20 && b[2]==0x00)
            Console.WriteLine("P2P napster login");
    }
}
```



```

alert tcp $HOME_NET any -> $EXTERNAL_NET 80
(msg:"MULTIMEDIA Quicktime User Agent access"; flow:to_server,established;
content:"User-Agent\: Quicktime";
classtype:policy-violation; sid:1436; rev:2;)

```



```

GET /napierstream
HTTP/1.0 User-Agent: QuickTime/7.1 (qtver=7.1;os=Windows NT 5.1)
Accept: application/x-rtsp-tunnelled
Pragma: no-cache Cache-Control: no-cache

```

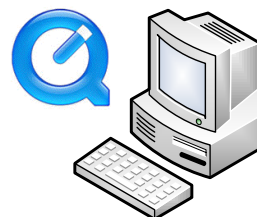
```

private static void device_PcapOnPacketArrival(..)
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)packet;
        int destPort = tcp.SourcePort;
        byte [] b = tcp.Data;
        ASCIIEncoding format = new ASCIIEncoding();
        string s = format.GetString(b); s=s.ToLower();
        if (destPort==80 && s.StartsWith("User-Agent\\\: Quicktime "))
            Console.WriteLine("MULTIMEDIA Quicktime User Agent access ");
    }
}

```



```
alert tcp $HOME
(msg:"MULTIMEDIA"
content:"User-Agent:QuickTime"
classtype:policy-violation)
```



```
private static
{
    if(packet
    {
        TCPpacket to
        int destPort
        byte [] b =
        ASCIIEncoding
        string s =
        if (destPort
        console.WriteLine
    }
}
```

(Untitled) - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: `http contains "GET" && ip.addr == 192.168.0.3` Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
295	13.610302	192.168.0.3	nwk-qtpix.apple.com	HTTP	GET /qtpix/qtpix_auto.mov HTTP/1.1
297	13.611322	192.168.0.3	nwk-qtpix.apple.com	HTTP	[TCP out-of-order] GET /qtpix/qtpix_auto.mov HTTP/1.1
400	15.735383	192.168.0.3	84.53.179.169	HTTP	GET /f/654/39/5m/qtpix.apple.com/f/654/39/5m/qtpix_auto.mov HTTP/1.1
402	15.736462	192.168.0.3	84.53.179.169	HTTP	[TCP out-of-order] GET /f/654/39/5m/qtpix_auto.mov HTTP/1.1

Frame 295 (248 bytes on wire, 248 bytes captured)

Ethernet II, Src: IntelCor\_34:02:f0 (00:15:00:34:02:f0), Dst: Gvc\_b7:5b:5a (00:c0:a8:b7:5b:5a)

Internet Protocol, Src: 192.168.0.3 (192.168.0.3), Dst: nwk-qtpix.apple.com (17.149.160.34)

Transmission Control Protocol, Src Port: 6213 (6213), Dst Port: http (80), Seq: 1, Ack: 1, Len: 194

Hypertext Transfer Protocol

GET /qtpix/qtpix\_auto.mov HTTP/1.1\r\n

Request Method: GET

Request URI: /qtpix/qtpix\_auto.mov

Request Version: HTTP/1.1

User-Agent: QuickTimeWinInet\r\n

Host: qtpix.apple.com\r\n

Cache-Control: no-cache\r\n

0000 00 c0 a8 b7 5b 5a 00 15 00 34 02 f0 08 00 45 00 ....[Z...4....E.

0010 00 ea 0f 70 40 00 80 06 78 3b c0 a8 00 03 11 95 ...p@...x;.....

0020 a0 22 18 45 00 50 c3 15 40 27 05 62 90 3e 50 18 ...E.P...@'.b.>P.

0030 44 70 1f f6 00 00 47 45 54 20 2f 71 74 70 69 78 dp....GE T /qtpix

0040 2f 71 74 70 69 78 5f 61 75 74 6f 2e 6d 6f 76 20 /qtpix\_a uto.mov

0050 48 54 54 50 2f 31 2e 31 0d 0a 55 73 65 72 2d 41 HTTP/1.1 ..User-A

0060 67 65 6e 74 3a 20 51 75 69 63 6b 54 69 6d 65 57 gent: Qu icktimew

0070 69 6e 49 6e 65 74 0d 0a 48 6f 73 74 3a 20 71 74 inInet.. Host: qt

0080 70 69 78 2e 61 70 70 6c 65 2e 63 6f 6d 0d 0a 43 pix.appl e.com..C

0090 61 63 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6e 6f ache-Con trol: no

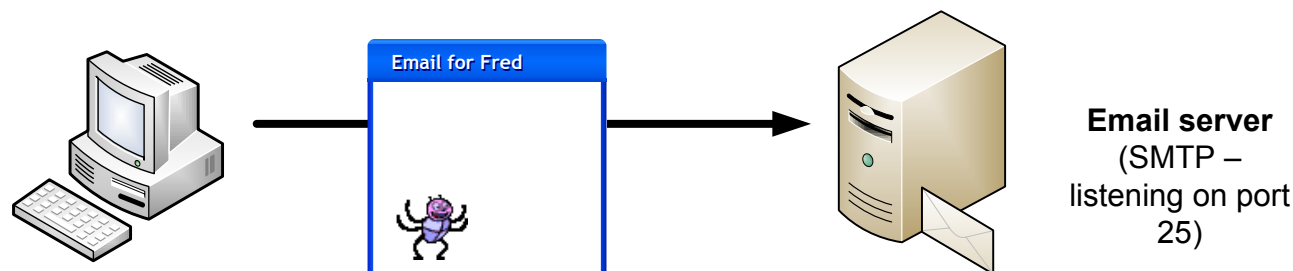
00a0 2d 63 61 63 68 65 0d 0a 43 6f 6f 6b 69 65 3a 20 -cache.. Cookie:

File: "C:\DOCUMENTS\WILLIAM\LOCALS~1\Temp\etherXXXa03428" 1448 KB 00:03:00 P: 4415 D: 4 M: 3 Drops: 0

```

alert tcp $SMTP_SERVERS any -> $EXTERNAL_NET 25
(msg:"VIRUS OUTBOUND .exe file attachment";
flow:to_server,established; content:"Content-Disposition|3a|";
content:"filename=|22|"; distance:0; within:30;
content:".exe|22|"; distance:0; within:30; nocase;
classtype:suspicious-filename-detect; sid:2160; rev:1;)

```



```

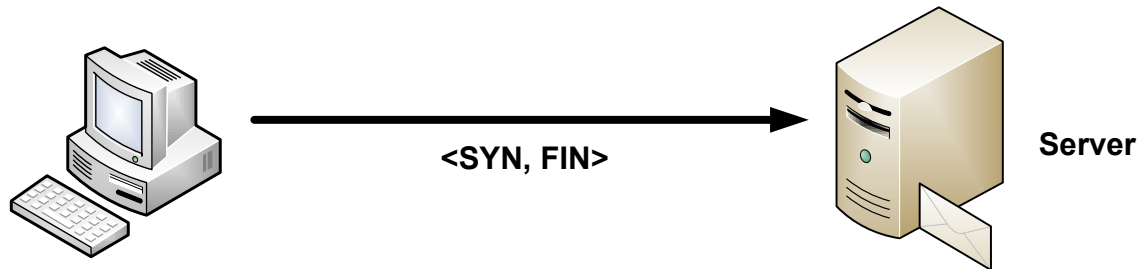
private static void device_PcapOnPacketArrival(..)
{
    TCPpacket tcp = (TCPpacket)packet;
    if(packet is TCPpacket)
    {
        int destPort = tcp.SourcePort;  byte [] b = tcp.Data;
        ASCIIEncoding format = new ASCIIEncoding();
        string s = format.GetString(b); s=s.ToLower();
        if (destPort==25 && s.IndexOf("Content-Disposition;")>0
            && s.IndexOf("filename=\"")>0 && s.IndexOf(".exe\"")>0 )
            Console.WriteLine("VIRUS OUTBOUND .exe file attachment");
    }
}

```





```
alert tcp any any -> any any (msg:"SYN FIN Scan"; flags: SF;)
alert tcp any any -> any any (msg:"FIN Scan"; flags: F;)
alert tcp any any -> any any (msg:"NULL Scan"; flags: 0;)
alert tcp any any -> any any (msg:"XMAS Scan"; flags: FPU;)
alert tcp any any -> any any (msg:"FULL XMAS Scan"; flags: SRAFPU;)
```



```
private static void device_PcapOnPacketArrival(...)
{
    if(packet is TCPpacket)
    {
        TCPpacket tcp = (TCPpacket)packet;
        if (tcp.Syn==true && tcp.Fin=true)
            Console.WriteLine("SYN FIN Scan");
    }
}
```



```

alert tcp $TELNET_SERVERS 23 -> $EXTERNAL_NET
any (msg:"TELNET root login";
flow:from_server,established;
content:"login|3A| root";
classtype:suspicious-login; sid:710; rev:7;)

```

(Untitled) - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: (ip.addr eq 192.168.0.4 and ip.addr eq 146.176.165.229) Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
2069	77.297871	pc165229.napier.ac.uk	192.168.0.4	TCP	telnet > 2403 [ACK] Seq=41 Ack=41 win=5840 Len=0
2070	77.298170	pc165229.napier.ac.uk	192.168.0.4	TELNET	Telnet Data ...
2071	77.298329	192.168.0.4	pc165229.napier.ac.uk	TELNET	Telnet Data ...
2073	77.386208	pc165229.napier.ac.uk	192.168.0.4	TCP	telnet > 2403 [ACK] Seq=53 Ack=44 win=5840 Len=0
2074	77.386276	192.168.0.4	pc165229.napier.ac.uk	TELNET	Telnet Data ...
2077	77.438728	pc165229.napier.ac.uk	192.168.0.4	TCP	telnet > 2403 [ACK] Seq=53 Ack=53 win=5840 Len=0
2078	77.439196	pc165229.napier.ac.uk	192.168.0.4	TELNET	Telnet Data ...
2079	77.439345	192.168.0.4	pc165229.napier.ac.uk	TELNET	Telnet Data ...
2084	77.491337	pc165229.napier.ac.uk	192.168.0.4	TELNET	Telnet Data ...
2085	77.491427	192.168.0.4	pc165229.napier.ac.uk	TELNET	Telnet Data ...

Frame 2084 (142 bytes on wire, 142 bytes captured)

Ethernet II, Src: Netgear\_b0:d6:8c (00:18:4d:b0:d6:8c), Dst: IntelCor\_34:02:f0 (00:15:00:34:02:f0)

Internet Protocol, Src: pc165229.napier.ac.uk (146.176.165.229), Dst: 192.168.0.4 (192.168.0.4)

Transmission Control Protocol, Src Port: telnet (23), Dst Port: 2403 (2403), Seq: 59, Ack: 56, Len: 88

Telnet

Data: \r\n

Data: Please login to NETLAB device.\r\n

Data: Unauthorized access is prohibited.\r\n

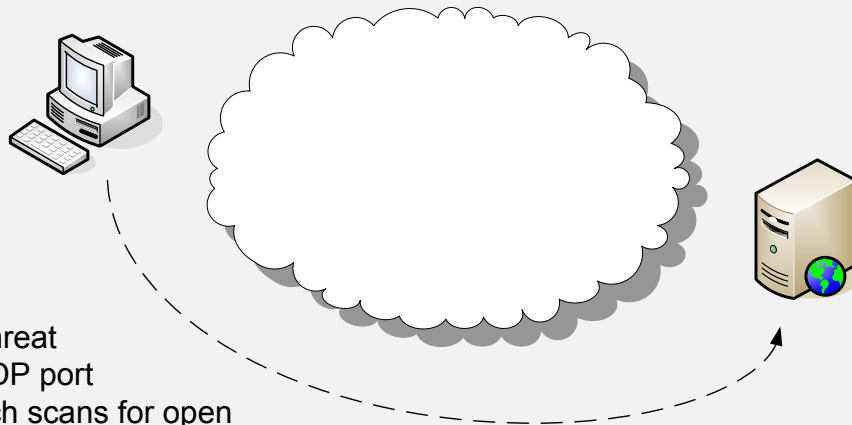
Data: \r\n

Data: NETLAB user ID:

0000	00 15 00 34 02 f0 00 18 4d b0 d6 8c 08 00 45 00	...4.... M.....E.
0010	00 80 b1 c2 40 00 34 06 9b 73 92 b0 a5 e5 c0 a8	....@.4. .S.....
0020	00 04 00 17 09 63 5a 60 4f bc 63 b2 ec 4f 50 18	....c2' o.c..OP.
0030	16 d0 aa 74 00 00 0d 0a 50 6c 65 61 73 65 20 6c	...t.... Please l
0040	6f 67 69 6e 20 74 6f 20 4e 45 54 4c 41 42 20 64	ogin to NETLAB d
0050	65 76 69 63 65 2e 0d 0a 55 6e 61 75 74 68 6f 72	evice... Unauthor
0060	69 7a 65 64 20 61 63 63 65 73 73 20 69 73 20 70	ized acc ess is p
0070	72 6f 68 69 62 69 74 65 64 2e 0d 0a 0d 0a 4e 45	rohibite d.....NE
0080	54 4c 41 42 20 75 73 65 72 20 49 44 3a 20	TLAB use r ID:

File: "C:\DOCUME~1\WILLIA~1\LOCAL5~1\Temp\etherXXXa02060" 3237 KB 00:01:57 P: 4729 D: 64 M: 0 Drops: 0

Open port 10?  
Open port 11?  
..  
Open port 8888?



A particular threat is the TCP/UDP port scanner, which scans for open ports on a host.

If an intruder finds one, it may try and connect to it.

### Typical scans:

Ping sweeps.

**TCP scans.**

**UDP scans.**

OS identification scans.

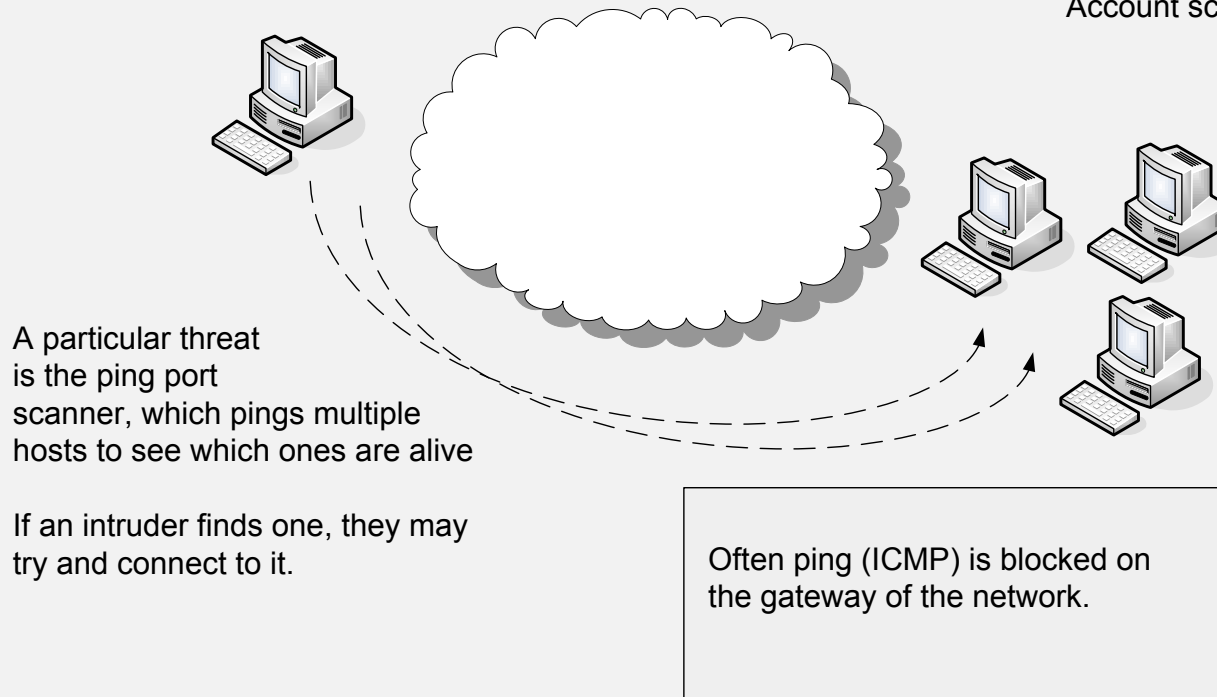
Account scans.

An open port is in the LISTEN state.

```
C:\log>netstat -a
Active Connections
Proto Local Address           Foreign Address         State
TCP    bills:epmap              bills:0                 LISTENING
TCP    bills:microsoft-ds       bills:0                 LISTENING
TCP    bills:1035               bills:0                 LISTENING
TCP    bills:3389               bills:0                 LISTENING
```

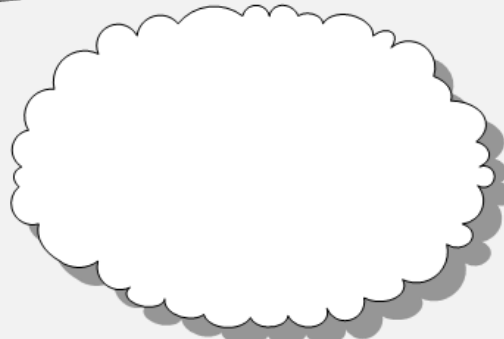
Ping 192.168.0.1?  
Ping 192.168.0.1?  
..  
Ping 192.168.0.253?  
Ping 192.168.0.254?

**Typical scans:**  
**Ping sweeps.**  
TCP scans.  
UDP scans.  
OS identification scans.  
Account scans.





TSeq(Class=RI%gcd=<8%SI=<2959A&>356%IPID=I)  
 T1(DF=Y%W=FAF0|402E%ACK=S++%Flags=AS%Ops=MNWNNT)  
 T2(Resp=N)  
 T3(Resp=N)  
 T4(DF=N%W=0%ACK=0%Flags=R%Ops=)  
 T5(DF=N%W=0%ACK=S++%Flags=AR%Ops=)  
 T6(DF=N%W=0%ACK=0%Flags=R%Ops=)  
 T7(Resp=N)  
 PU(DF=N%TOS=0%IPLen=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)



**TSeq. This is where SYN packets are sent, and the TCP sequence numbers are analysed.**

T1. This is a SYN packet with certain options (WNMTE) set is sent to an open TCP port.

T2. This is a NULL packet with options (WNMTE) and is sent to an open TCP port.

T3. This is a SYN,FIN,PSH,URG packet with options (WNMTE), and sent to an open TCP port.

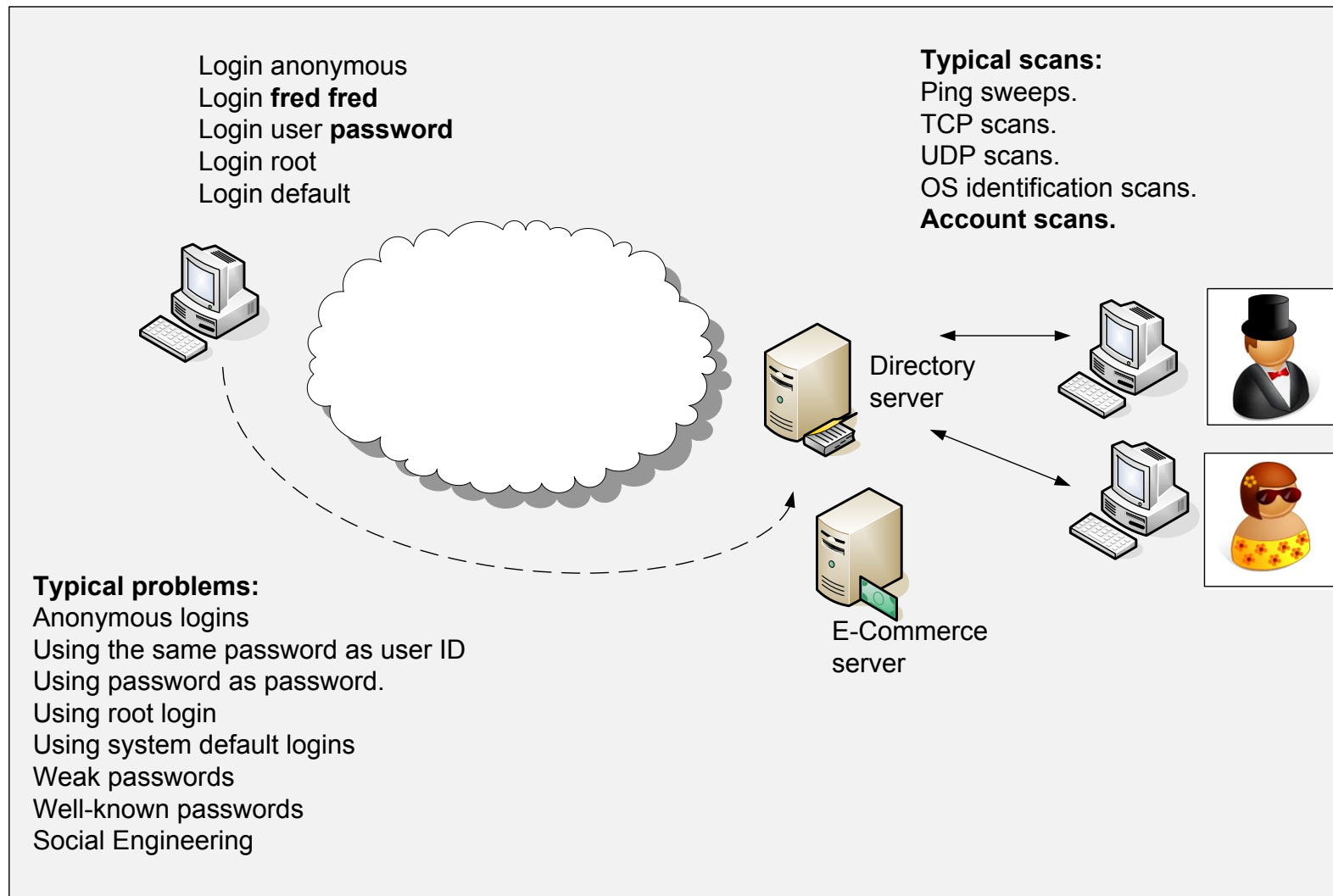
T4. This is an ACK packet with options (WNMTE) and is sent to an open TCP port.

T5. This is a SYN packet with options (WNMTE) and is sent to a closed TCP port.

T6. This is an ACK packet with options (WNMTE) and is sent to a closed TCP port.

T7. This is a FIN,PSH,URG packet with options (WNMTE) and is sent to a closed TCP port.

PU. This is a packet sent to a closed UDP port.



seg8 - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: `tcp.port==21 && ip.addr==10.0.1.16` Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
652	6.879856	128.129.31.13	10.0.1.16	FTP	Request: USER admin
657	6.883046	10.0.1.16	128.129.31.13	FTP	Response: 331 Password re
658	6.883822	128.129.31.13	10.0.1.16	FTP	Request: PASS marietta
669	6.970991	10.0.1.16	128.129.31.13	TCP	ftp > 4197 [ACK] Seq=193
670	6.980873	10.0.1.16	128.129.31.13	TCP	ftp > 4198 [ACK] Seq=193
689	7.095867	128.129.31.13	10.0.1.16	FTP	Request: USER admin
690	7.100531	10.0.1.16	128.129.31.13	FTP	Response: 331 Password re
694	7.101144	128.129.31.13	10.0.1.16	TCP	4200 > ftp [ACK] Seq=13 A
695	7.101865	128.129.31.13	10.0.1.16	FTP	Request: PASS lorin
698	7.103811	128.129.31.13	10.0.1.16	FTP	Request: USER admin
699	7.108498	10.0.1.16	128.129.31.13	FTP	Response: 331 Password re
700	7.108912	128.129.31.13	10.0.1.16	TCP	4201 > ftp [ACK] Seq=13 A
701	7.109803	128.129.31.13	10.0.1.16	FTP	Request: PASS lorraine
702	7.110779	128.129.31.13	10.0.1.16	FTP	Request: USER admin
703	7.115535	10.0.1.16	128.129.31.13	FTP	Response: 331 Password re
704	7.116144	128.129.31.13	10.0.1.16	TCP	4202 > ftp [ACK] Seq=13 A
705	7.116858	128.129.31.13	10.0.1.16	FTP	Request: PASS louis
707	7.121790	128.129.31.13	10.0.1.16	FTP	Request: USER admin
708	7.126628	10.0.1.16	128.129.31.13	FTP	Response: 331 Password re
709	7.127067	128.129.31.13	10.0.1.16	TCP	4203 > ftp [ACK] Seq=13 A
710	7.127783	128.129.31.13	10.0.1.16	FTP	Request: PASS love
711	7.129812	128.129.31.13	10.0.1.16	FTP	Request: USER admin

Frame 711 (78 bytes on wire (78 bytes captured))

0000 00 01 02 a0 f2 d3 00 0c ce 85 ab 60 08 00 45 00 .....E.  
0010 00 40 9b eb 40 00 3f 06 f5 2e 80 81 1f 0d 0a 00 .@..@.?.....  
0020 01 10 10 6c 00 15 e2 97 ad a6 a0 cc 42 18 80 18 ...1....B...

IDS

FTP Username/  
Password scan

User account scans



seg8 - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: `tcp.port==21 && ip.addr==10.0.1.16` Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
652	6.879856	128.129.31.13	10.0.1.16	FTP	Request: USER admin
657	6.883046	10.0.1.16	128.129.31.13	FTP	Response: 331 Password required
658	6.883822	128.129.31.13	10.0.1.16	FTP	Request: PASS marietta

seg8 - Wireshark

File Edit View Go Capture Analyze Statistics Help

Filter: `tcp.port==21 && ip.addr==10.0.1.16` Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
94650	1149.666599	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
94768	1151.196578	128.129.31.13	10.0.1.16	FTP	Request: USER admin
94770	1151.201368	10.0.1.16	128.129.31.13	FTP	Response: 331 Password required
94773	1151.251146	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95017	1151.251146	10.0.1.16	10.0.1.16	FTP	Request: PASS password
95019	1151.251146	128.129.31.13	10.0.1.16	FTP	Response: 230 User admin logged in
95025	1151.251146	10.0.1.16	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95026	1151.251146	10.0.1.16	10.0.1.16	FTP	Request: SYST
95027	1153.453845	10.0.1.16	128.129.31.13	FTP	Response: 215 UNIX Type: L
95030	1153.503236	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95200	1155.190379	128.129.31.13	10.0.1.16	FTP	Request: PWD
95201	1155.190903	10.0.1.16	128.129.31.13	FTP	Response: 257 "/usr/home/
95202	1155.191353	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95410	1157.539997	128.129.31.13	10.0.1.16	FTP	Request: CWD ..
95411	1157.540894	10.0.1.16	128.129.31.13	FTP	Response: 250 CWD command
95412	1157.541410	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95633	1159.491580	128.129.31.13	10.0.1.16	FTP	Request: PWD
95634	1159.492028	10.0.1.16	128.129.31.13	FTP	Response: 257 "/usr/home/
95635	1159.492642	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=
95780	1161.278380	128.129.31.13	10.0.1.16	FTP	Request: CWD ..
95781	1161.278939	10.0.1.16	128.129.31.13	FTP	Response: 250 CWD command
95782	1161.279441	128.129.31.13	10.0.1.16	TCP	owserver > ftp [ACK] Seq=

Successful login to FTP server

Frame 94768: 678 bytes captured on interface 0

0000 00 01 02 a0 f2 d3 00 0c ce 85 ab 60 08 00 45 10 .....E.

0010 00 40 c7 48 40 00 3f 06 c9 c1 80 81 1f 0d 0a 00 .@.H@.?.....

0020 01 10 10 d0 00 15 2a e9 6b fd 06 1f cc 52 80 18 .....\*.k....R..

0030 05 54 88 35 00 00 01 01 08 0e 00 7a b3 52 00 00 .....



## SCAN.RULE

```
preprocessor flow: stats_interval 0 hash 2
preprocessor sfportscan: proto { all } scan_type { all }
                        sense_level { low } logfile { portscan.log }
```

## PORTSCAN.LOG

```
C:\> snort -c scan.rule -dev -i 3 -p -l c:\\bill -K ascii
```

Initializing Preprocessors!

Initializing Plug-ins!

Parsing Rules file scan.rule

```
-----[Flow Config]-----
| Stats Interval: 0
| Hash Method:    2
| Memcap:         10485760
| Rows :         4096
| Overhead Bytes: 16388(%0.16)
-----
```

Portscan Detection Config:

```
Detect Protocols:  TCP UDP ICMP IP
Detect Scan Type:  portscan portsweep decoy_portscan distributed_portscan
Sensitivity Level: Low
Memcap (in bytes): 1048576
Number of Nodes:   3869
Logfile:           c:\\bill\\portscan.log
```

Tagged Packet Limit: 256

...

```
C:\>nmap -o -A 192.168.0.1
```

Starting Nmap 4.20 ( <http://insecure.org> ) at 2007-01-09 21:58 GMT Standard Time

Interesting ports on 192.168.0.1:

Not shown: 1695 closed ports

PORT	STATE	SERVICE
80/tcp	open	http
8888/tcp	open	sun-answerbook

MAC Address: 00:0B:44:F5:33:D5 (The Linksys Group)

Nmap finished: 1 IP address (1 host up) scanned in 1.500 seconds

Time: 08/17-14:41:54.495296

event\_ref: 0

192.168.0.3 -> 64.13.134.49 (portscan) TCP Portsweep

Priority Count: 5

Connection Count: 135

IP Count: 43

Scanned IP Range: 64.13.134.49:216.239.59.99

Port/Proto Count: 1

Port/Proto Range: 80:80

Time: 08/17-14:42:52.431092

event\_ref: 0

192.168.0.3 -> 192.168.0.1 (portscan) TCP Portsweep

Priority Count: 5

Connection Count: 10

IP Count: 5

Scanned IP Range: 66.249.93.165:192.168.0.7

Port/Proto Count: 3

Port/Proto Range: 80:2869

Time: 08/17-14:42:52.434852

event\_ref: 0

192.168.0.3 -> 192.168.0.1 (portscan) TCP Portscan

Priority Count: 5

Connection Count: 9

IP Count: 1

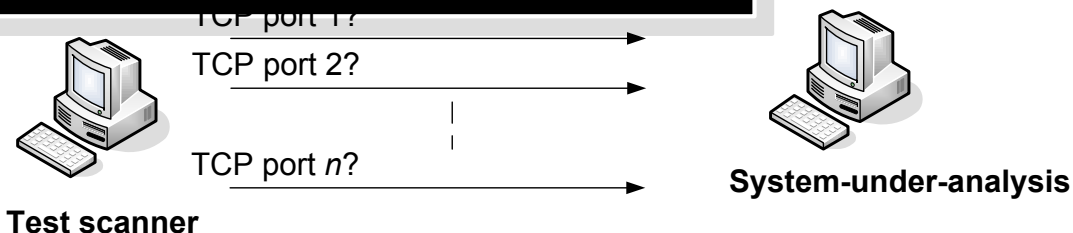
Scanner IP Range: 192.168.0.3:192.168.0.3

Port/Proto Count: 10

Port/Proto Range: 21:636

## NOTE:

The NMAP program should only be used on machines which you are under control of, and in a local, and isolated environment. It should only be used to determine possible weaknesses and vulnerabilities.



**Port scanner pre-processor**

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions


User profiling

Honeypots


IPS

Conclusions







**Name:** Fiona Smith  
**Nationality:** British  
**Location:** Edinburgh  
**Gender:** Female  
**Typical purchase:** Computer equipment  
**Average Purchases/week:** 5  
**Average Value of purchases:** £30  
**Browser used:** Mozilla  
**Date of last purchase:** 6 May 2008  
**Email address:** f.smith@nowhere




**Name:** Fred McLean  
**Nationality:** USA  
**Location:** Washington  
**Gender:** Male  
**Typical purchase:** Fish Food  
**Average Purchases/week:** 50  
**Average Value of purchases:** \$4  
**Browser used:** IE  
**Date of last purchase:** 18 Sept 2008  
**Email address:** f.mclean@usa



**Name:** Michel Weber  
**Nationality:** German  
**Location:** Munich  
**Gender:** Male  
**Typical purchase:** Flowers  
**Average Purchases/week:** 0.005  
**Average Value of purchases:** €43  
**Browser used:** Opera  
**Date of last purchase:** 1 Mar 2007  
**Email address:** m\_weber@de



**Name:** Amélie Cheney  
**Nationality:** French  
**Location:** Paris  
**Gender:** Female  
**Typical purchase:** Clothes  
**Average Purchases/week:** 70  
**Average Value of purchases:** €13  
**Browser used:** Mozilla  
**Date of last purchase:** 16 Sept 2008  
**Email address:** a.cheney@fr.edu



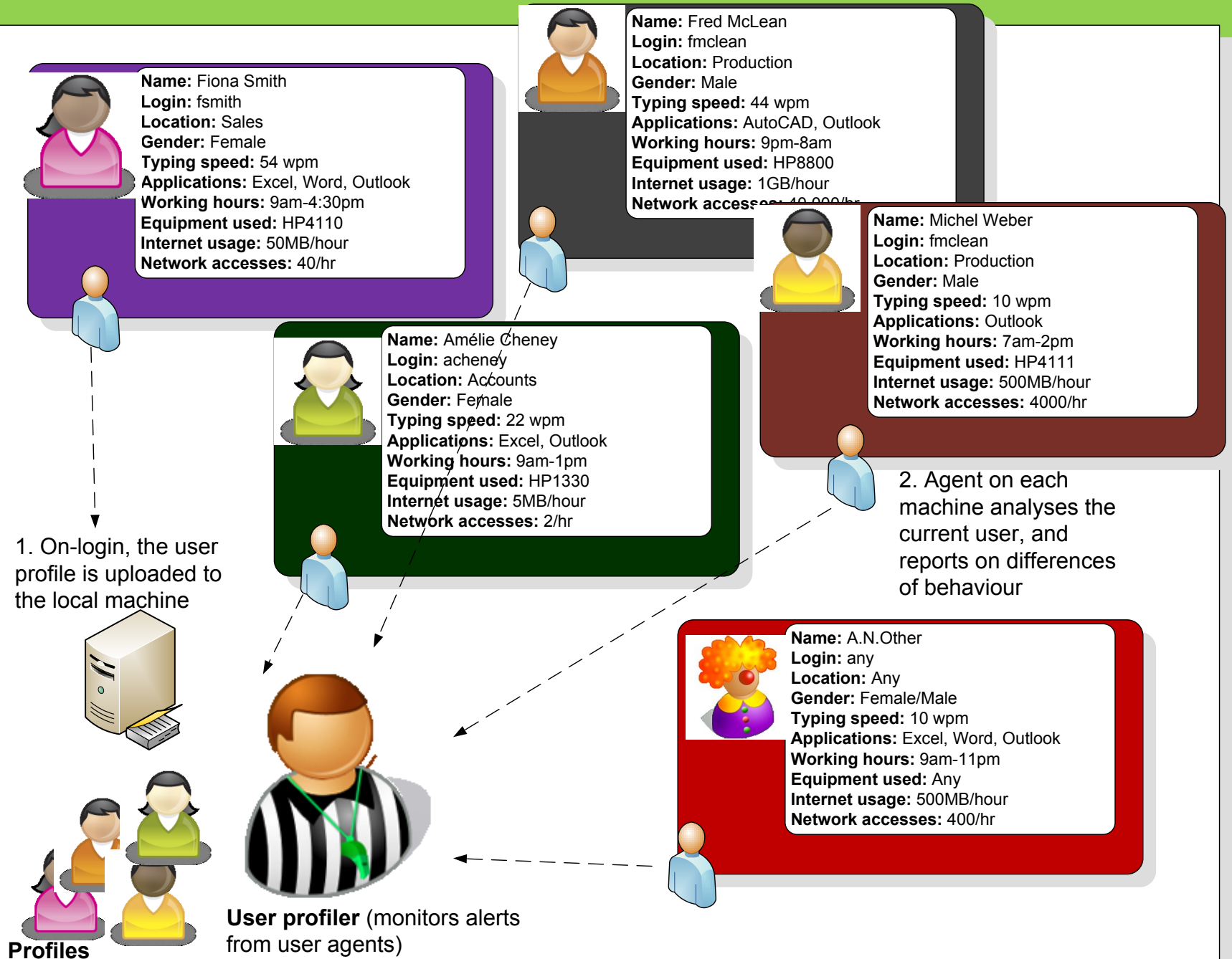
**Name:** A.N.Other  
**Nationality:** Any  
**Location:** Nowhere  
**Gender:** Female/Male  
**Typical purchase:** High-value goods  
**Average Purchases/week:** 1000  
**Average Value of purchases:** \$9999  
**Browser used:** Not known  
**Date of last purchase:** Today  
**Email address:** doesnt@exist



**User profiler** (such as bank transaction agent)

Transactions are checked against user profile

User/behaviour profiling is especially useful in fraud detection



# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

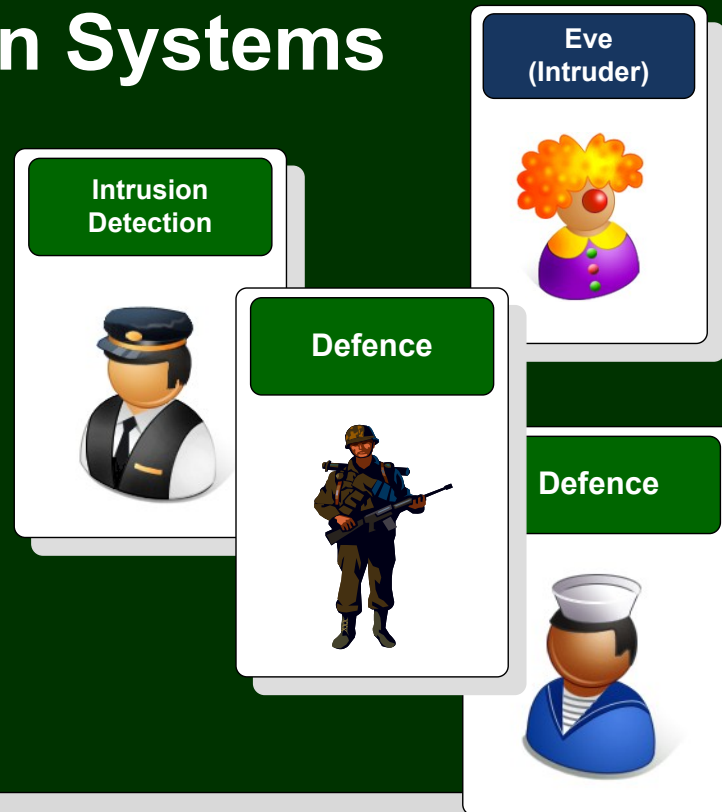
A few intrusions

User profiling

Honeypots

IPS

Conclusions



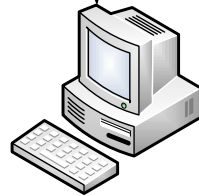
Honeypots

This device has all the required weaknesses, such as:

- Default administrator/password.
- Dummy users with weak passwords.
- Ports open for connection.
- React to virus/worm systems (but simulate conditions).



**Intruder**



**Honeypot**

**Servers/  
systems**



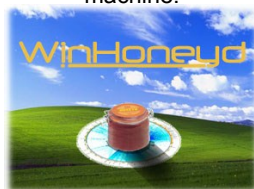


**High-interaction honeypot.** This simulates all the aspects of the operating system

**Open ports:** 110 (POP-3), 80 (HTTP), 21 (FTP), 22 (SSH)



**Low-interaction honeypot.** This simulates only part of the network stack (such as for **Honeyd**)  
- can be virtual (from a virtual machine) or simulated by another machine.



### Honeyd.conf

```
create default
set default personality "windows xp"
set default default tcp action reset
add default tcp port 110 "sh scripts/pop.sh"
add default tcp port 80 "perl scripts/iis-0.95/main.pl"
add default tcp port 25 block
add default tcp port 21 "sh scripts/ftp.sh"
add default tcp port 22 proxy $ipsrc:22
add default udp port 139 drop
set default uptime 3284460
```

```
### Cisco router
create router
set router personality "Cisco PIX Firewall (PixOS 5.2 - 6.1)"
add router tcp port 23 "/usr/bin/perl scripts/router-telnet.pl"
set router default tcp action reset
set router uid 32767 gid 32767
set router uptime 1327650
# Bind specific templates to specific IP address
# If not bound, default to windows template
bind 192.168.1.150 router
```

```

# Copyright 2002 Niels Provos <provos@citi.umich.edu>
# All rights reserved.
# For the license refer to the main source code of Honeyd.
# Don't echo Will Echo Will Surpress Go Ahead
$return = pack('cccccccc', 255, 254, 1, 255, 251, 1, 255, 251, 3);
syswrite STDOUT, $return, 9;

$string =
"Users (authorized or unauthorized) have no explicit or\r
implicit expectation of privacy. Any or all uses of this\r
system may be intercepted, monitored, recorded, copied,\r
audited, inspected, and disclosed to authorized site,\r
and law enforcement personnel, as well as to authorized\r
officials of other agencies, both domestic and foreign.\r
By using this system, the user consents to such\r
interception, monitoring, recording, copying, auditing,\r
inspection, and disclosure at the discretion of authorized\r
site.\r
\r
Unauthorized or improper use of this system may result in\r
administrative disciplinary action and civil and criminal\r
penalties. By continuing to use this system you indicate\r
your awareness of and consent to these terms and conditions\r
of use. LOG OFF IMMEDIATELY if you do not agree to the\r
conditions stated in this warning.\r
\r
\r
\r
User Access Verification\r
";

syswrite STDOUT, $string;
$count = 0;
while ($count < 3) {
    do {
        $count++;
        syswrite STDOUT, "\r\n";
        $word = read_word("Username: ", 1);
    } while (!$word && $count < 3);
    if ($count >= 3 && !$word) {
        exit;
    }
    $password = read_word("Password: ", 0);
    if (!$password) {
        syswrite STDOUT, "% Login invalid\r\n";
    } else {
        syswrite STDERR, "Attempted login: $word/$password";
        syswrite STDOUT, "% Access denied\r\n";
    }
}
}

```

```

exit;
sub read_word {
    local $prompt = shift;
    local $echo = shift;
    local $word;

    syswrite STDOUT, "$prompt";

    $word = "";
    $alarmed = 0;
    eval {
        local $$SIG{ALRM} = sub { $alarmed = 1; die; };
        alarm 30;
        $finished = 0;
        do {
            $nread = sysread STDIN, $buffer, 1;
            die unless $nread;
            if (ord($buffer) == 0) {
                ; #ignore
            } elsif (ord($buffer) == 255) {
                sysread STDIN, $buffer, 2;
            } elsif (ord($buffer) == 13 || ord($buffer) == 10) {
                syswrite STDOUT, "\r\n" if $echo;
                $finished = 1;
            } else {
                syswrite STDOUT, $buffer, 1 if $echo;
                $word = $word.$buffer;
            }
        } while (!$finished);
        alarm 0;
    };
    syswrite STDOUT, "\r\n" if $alarmed || !$echo;
    if ($alarmed) {
        syswrite STDOUT, "% $prompt timeout expired!\r\n";
        return (0);
    }

    return ($word);
}

```



# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

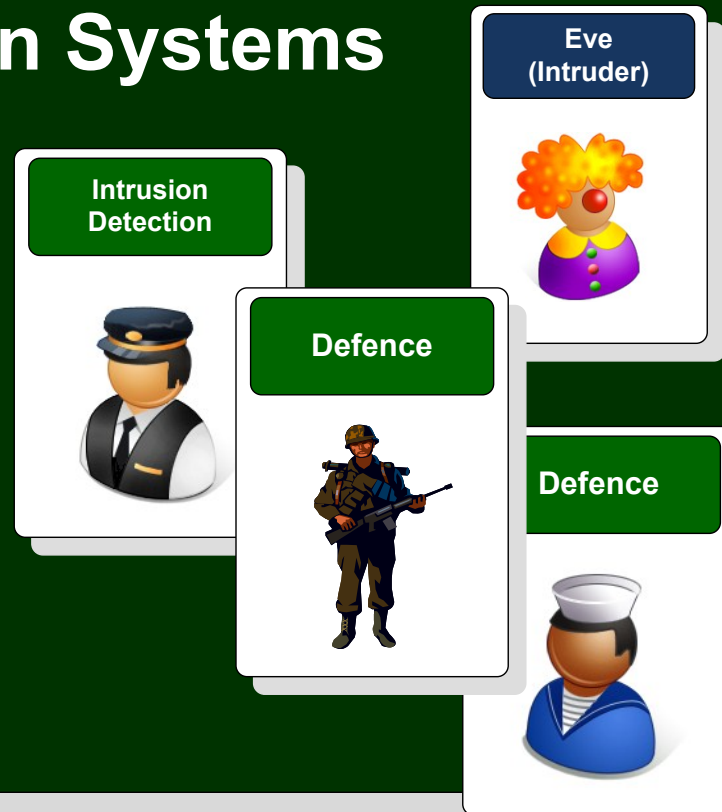
A few intrusions

User profiling

Honeypots

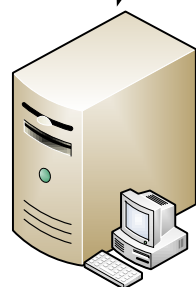
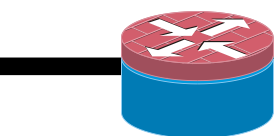
IPS

Conclusions



IPS and In/out-line

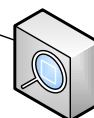
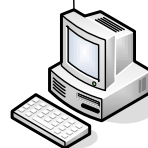
**In-line IDS, which can decide to drop a packet, alarm (send an alert/log) or reset a connection.**



**Syslog Server**

(stores alerts/logs/etc)

**Out-of-line IDS, which passively listens to traffic and cannot actually drop packets (unless there is an IPS)**



Intrusion Detection System

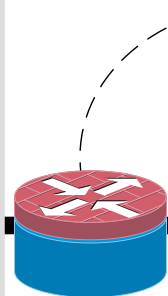


**In-line IDS** has the advantage that they can act on the intrusion, but it has a performance impact. The signatures are also difficult to change/upgrade.

**Out-of-line IDS** has the advantage of being able to more easily craft an IDS rule, but cannot take actions, directly.

## Example Cisco IDS signatures

**1001** – Bad IP Options  
(Info)  
**1100** – IP Fragment (Attack)  
**2000** – ICMP echo reply  
(Info)  
**2154** – Ping of death  
(Attack)  
**3041** – SYN/FIN Packet  
(Attack)  
**3040** – NULL TCP Packet  
(Attack)  
**3050** – Half open SYN  
(Attack)  
**3152** – CWD Root on FTP  
(Info)



**Syslog  
Server**  
(stores  
alerts/logs/  
etc)

```
(config)# ip audit ?
  attack      Specify default action for attack signatures
  info        Specify default action for informational signatures
  name        Specify an IDS audit rule
  notify      Specify the notification mechanisms (nr-director or log) for the
              alarms
  po          Specify nr-director's PostOffice information (for sending events
              to the nr-directors)
  signature    Add a policy to a signature
  smtp        Specify SMTP Mail spam threshold
(config)# ip audit notify ?
  log          Send events as syslog messages
  nr-director  Send events to the nr-director
(config)# ip audit notify log
(config)# logging 132.191.125.3
(config)# ip audit ?
  attack      Specify default action for attack signatures
  info        Specify default action for informational signatures
  name        Specify an IDS audit rule
  notify      Specify the notification mechanisms (nr-director or log) for the
              alarms
  po          Specify nr-director's PostOffice information (for sending events
              to the nr-directors)
  signature    Add a policy to a signature
  smtp        Specify SMTP Mail spam threshold
(config)# ip audit info ?
  action      Specify the actions
(config)# ip audit info action ?
  alarm       Generate events for matching signatures
  drop        Drop packets matching signatures
  reset       Reset the connection (if applicable)
(config)# ip audit info action drop
(config)# ip audit attack action reset
(config)# ip audit signature ?
  <1-65535>    Signature to be configured
(config)# ip audit signature 1005 disable
(config)# ip audit smtp ?
  spam        Specify the threshold for spam signature
  <cr>
(config)# ip audit smtp spam ?
  <1-65535>    Threshold of correspondents to trigger alarm
(config)# ip audit smtp spam 4
```

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

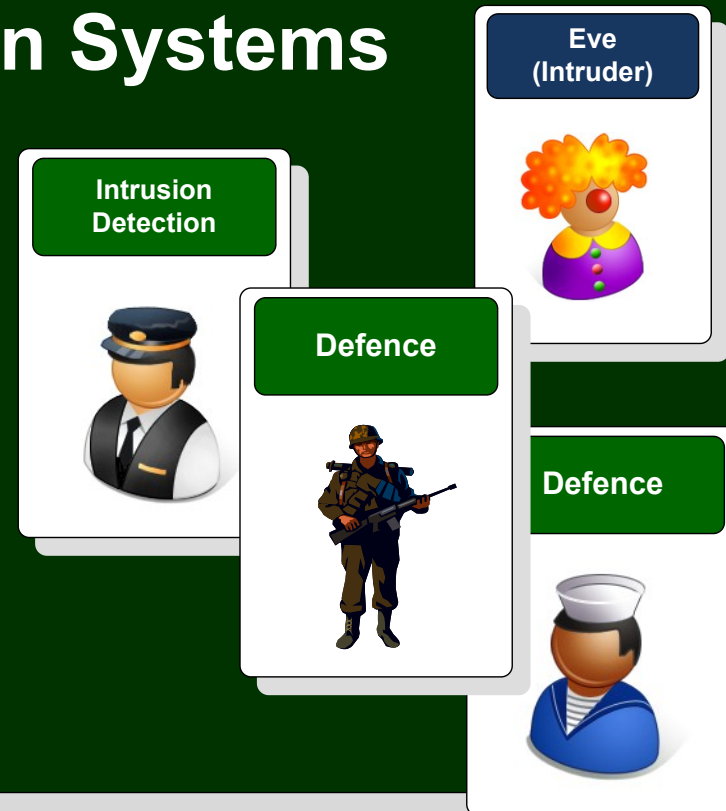
A few intrusions

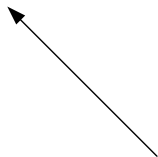
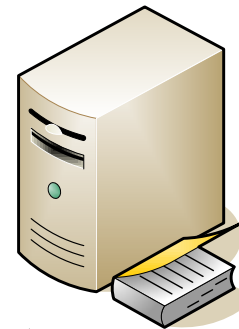
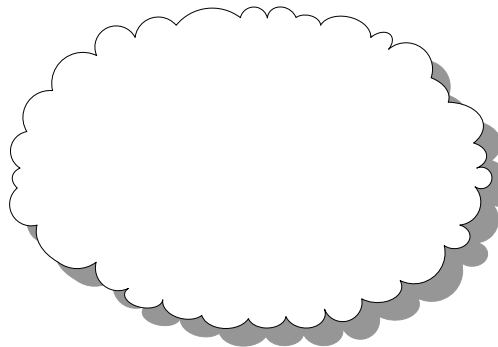
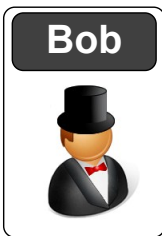
User profiling

Honeypots

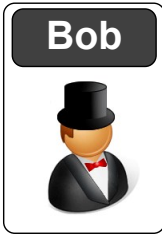
IPS

Conclusions



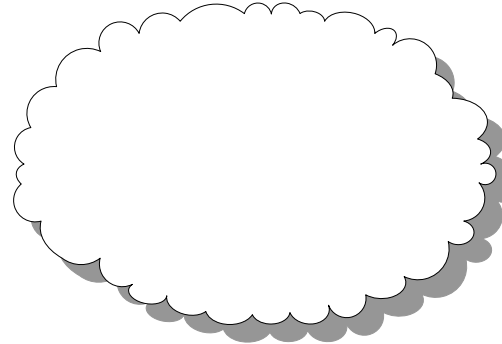


**Anomaly detection:**  
Learn normal activity, such as:  
User activity.  
System activity  
Server activity  
Network activity  
Application activity  
And so on



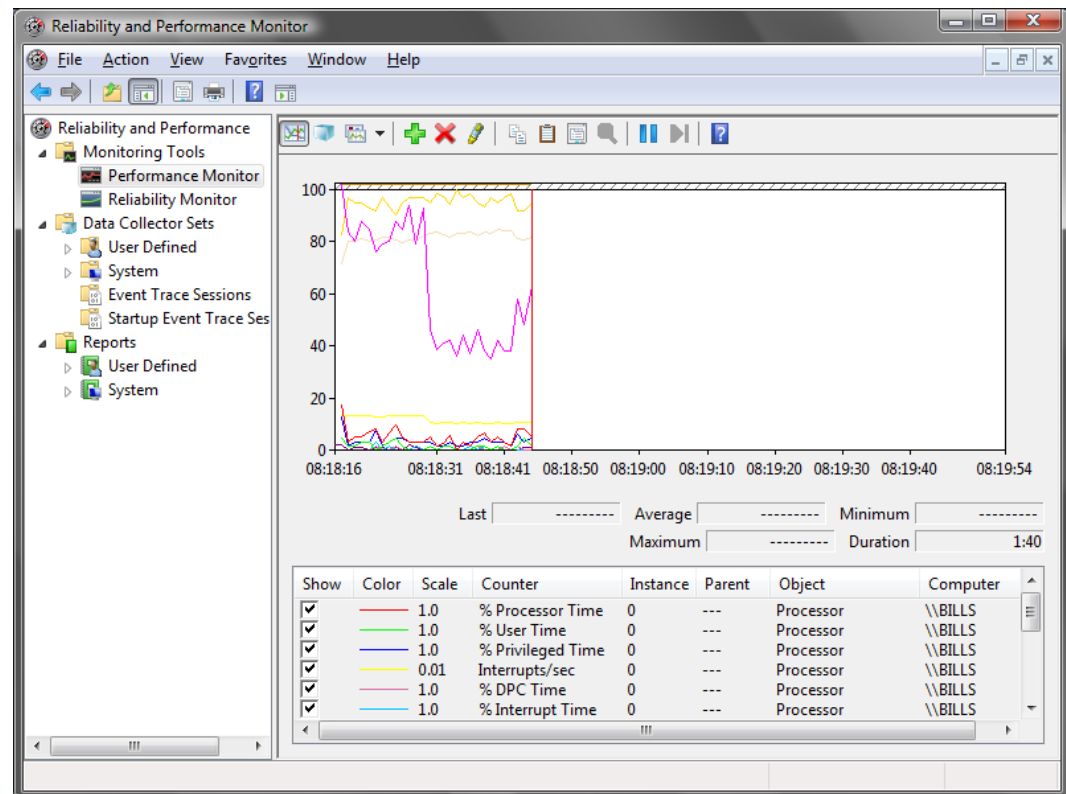
### User anomaly:

Typing speed  
Packages used  
Working hours  
Emails sent/hr  
Web sites visited

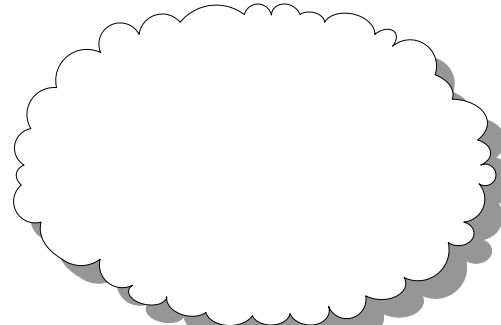


### System anomaly:

CPU Usage/min  
Threads/min  
Disk writes/min



Bob



## Network anomaly

IP packets (%)

TCP packets (%)

HTTP (%)

FTP (%)

FTP threshold (2%)

Ethereal: Protocol Hierarchy Statistics							
Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100.00%	1108	774129	0.021	0	0	0.000
Ethernet	100.00%	1108	774129	0.021	0	0	0.000
Logical-Link Control	1.17%	13	1174	0.000	0	0	0.000
Internetwork Packet eXchange	1.17%	13	1174	0.000	0	0	0.000
IPX Routing Information Protocol	0.54%	6	348	0.000	6	348	0.000
NetBIOS over IPX	0.54%	6	588	0.000	6	588	0.000
Name Management Protocol over IPX	0.09%	1	238	0.000	0	0	0.000
SMB (Server Message Block Protocol)	0.09%	1	238	0.000	0	0	0.000
SMB MailSlot Protocol	0.09%	1	238	0.000	0	0	0.000
Microsoft Windows Browser Protocol	0.09%	1	238	0.000	1	238	0.000
Internet Protocol	98.65%	1093	772853	0.021	0	0	0.000
User Datagram Protocol	14.44%	160	44326	0.001	0	0	0.000
Hypertext Transfer Protocol	9.03%	100	34560	0.001	100	34560	0.001
Domain Name Service	3.79%	42	6718	0.000	42	6718	0.000
Simple Network Management Protocol	0.81%	9	1635	0.000	9	1635	0.000
NetBIOS Name Service	0.45%	5	460	0.000	5	460	0.000
NetBIOS Datagram Service	0.36%	4	953	0.000	0	0	0.000
SMB (Server Message Block Protocol)	0.36%	4	953	0.000	0	0	0.000
SMB MailSlot Protocol	0.36%	4	953	0.000	0	0	0.000
Microsoft Windows Browser Protocol	0.36%	4	953	0.000	4	953	0.000
Transmission Control Protocol	83.48%	925	727935	0.019	583	353311	0.009
Hypertext Transfer Protocol	27.89%	309	368157	0.010	298	360076	0.010
Line-based text data	0.36%	4	3301	0.000	0	0	0.000
Media Type	0.36%	4	2668	0.000	0	0	0.000
CompuServe GIF	0.27%	3	2112	0.000	3	2112	0.000
SSH Protocol	2.98%	33	6467	0.000	33	6467	0.000
Internet Control Message Protocol	0.72%	8	592	0.000	8	592	0.000
Address Resolution Protocol	0.18%	2	102	0.000	2	102	0.000

Anomaly

IDS

User, system and network anomaly

# Intrusion Detection Systems

Introduction

Threats

Types

Host or Network?

Agent-based

Snort

A simple rule

A few intrusions

User profiling

Honeypots

IPS

Conclusions



Conclusions