CSCI262
System Security (**S6**)

Malicious code or malware

Outline

- What is malware?
- Types of malware.
  - Viruses.
  - Trojan horses.
  - Worms.
- Malware propagation.
- Malware detection.

What is malicious code (malware)?

- Malware or malicious code is computer code that is designed to modify computer systems without the consent of the owner or operator.
- Malicious mobile code is malware that is furthermore designed to move:
  - From computer to computer and network to network.

Dilbert: 29-Nov-2003

Some numbers

- This is the report on 2014, released in April 2015.

Attackers Are Moving Faster, Defenses Are Not

Within four hours of the Heartbleed vulnerability becoming public, Symantec saw a surge of attackers stepping up to exploit it. Reaction time has not increased at an equivalent pace. Advanced attackers continue to favor zero-day vulnerabilities to silently sneak onto victims' computers, and 2014 had an all-time high of 24 discovered zero-day vulnerabilities. As we observed with Heartbleed, attackers moved in to exploit these vulnerabilities much faster than vendors could create and roll out patches. In 2014, it took 204 days, 22 days, and 53 days, for vendors to provide a patch for the top three most exploited zero-day vulnerabilities. By comparison, the average time for a patch to be issued in 2013 was only four days. The most frightening part, however, is that the top five zero-days of 2014 were actively used by attackers for a combined 295 days before patches were available.

Some more numbers …

- Mobile:
  - “Symantec found that 17 percent of all Android apps (nearly one million total) were actually malware in disguise.”
  - “Grayware apps, which aren’t malicious by design but do annoying and inadvertently harmful things like track user behaviour, accounted for 36 percent of all mobile apps.”
  - The rate of innovation with mobile malware is slowing though …
- 317 million new malware variants, vs 252 million in 2013.

The best known malware types

- **Viruses**:
  - These infect files on the infected host, or in the boot area, to help aid replication.
- **Trojan horses or Trojans**:
  - These are non-replicating program that openly exhibit one desirable behaviour, it might be a game for example, but have some real intent hidden from the user. This real intent could, for example, be to open ports on a machine to allow attackers access.

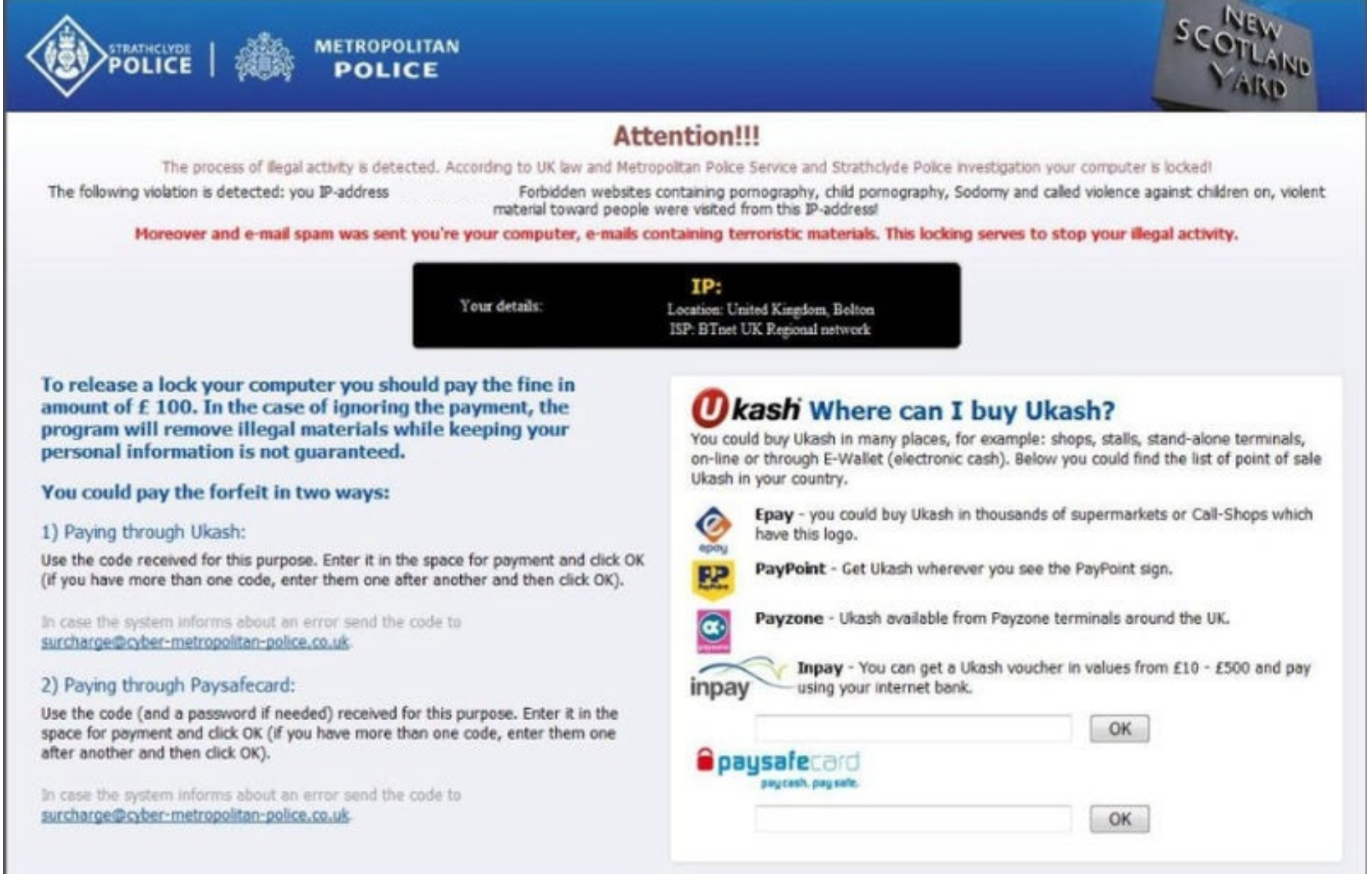
- **Worms**:
  - These replicate themselves to spread with minimal user interaction. Worms typically use widely available applications such as email to spread. Worms exploit holes in software and implementations.
  - Trojans are really making their own holes!

Other malware

- Backdoors/trapdoors: These bypass normal access control.
  - Some recent news: Apple vs FBI, Snowden report
- Logic bombs: Logic activated fragment which acts of breach security policies.
- Spyware: Collects information and transmits elsewhere.
  - This might sometimes be a special case of a Trojan.
- Flooders, Keyloggers, Zombie bot, …
- …

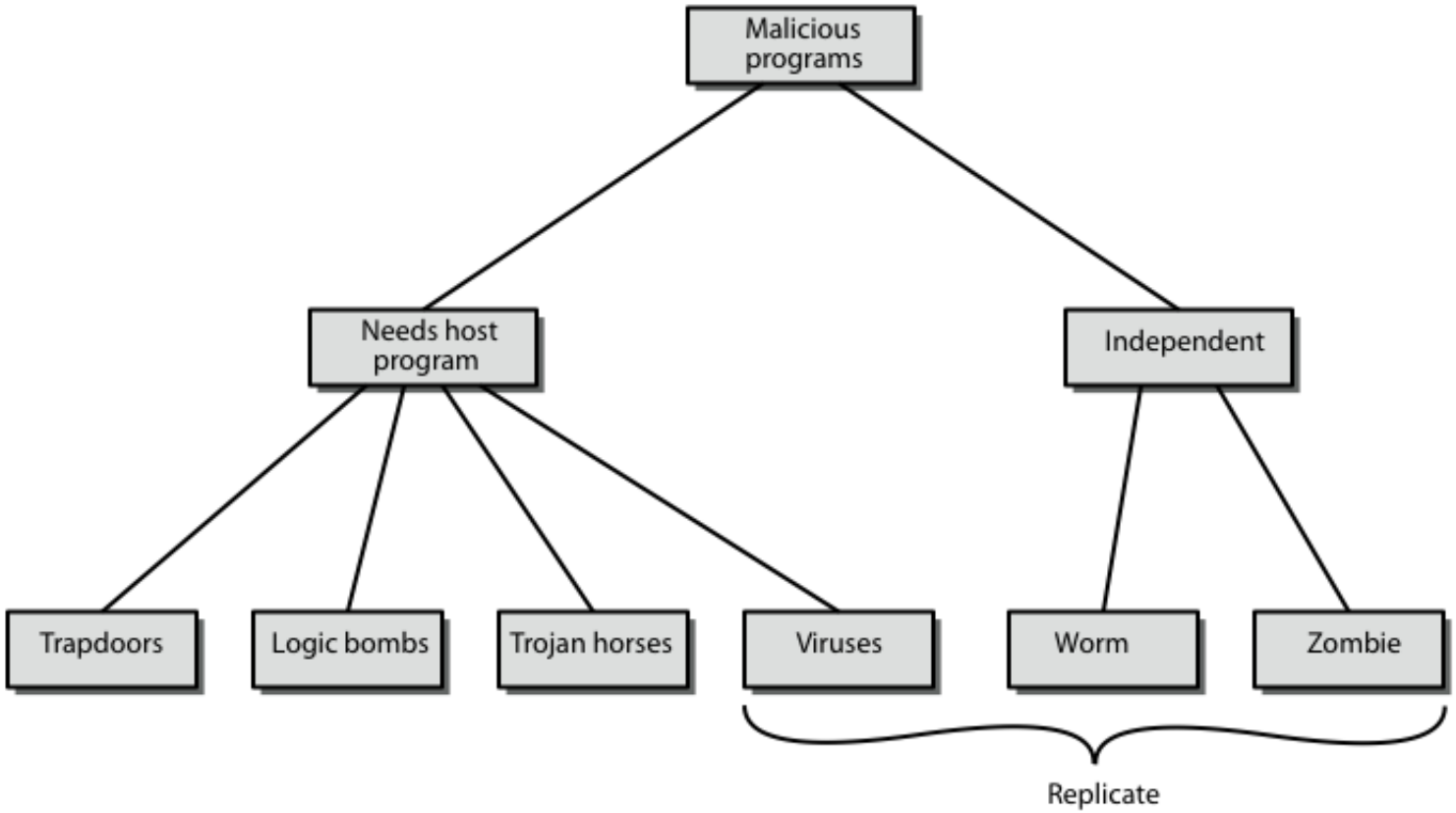
Ransomware

- From the Symantec report (April 2015)
- “Ransomware attacks grew 113 percent in 2014, along with 45 times more crypto-ransomware attacks.” ~ 24,000 per day…
- Randomware typically involves an attack on availability that will not be stopped until some payment is made.
  - Crypto-ransomware involves encrypting files.
  - When you pay, you get the password/key.

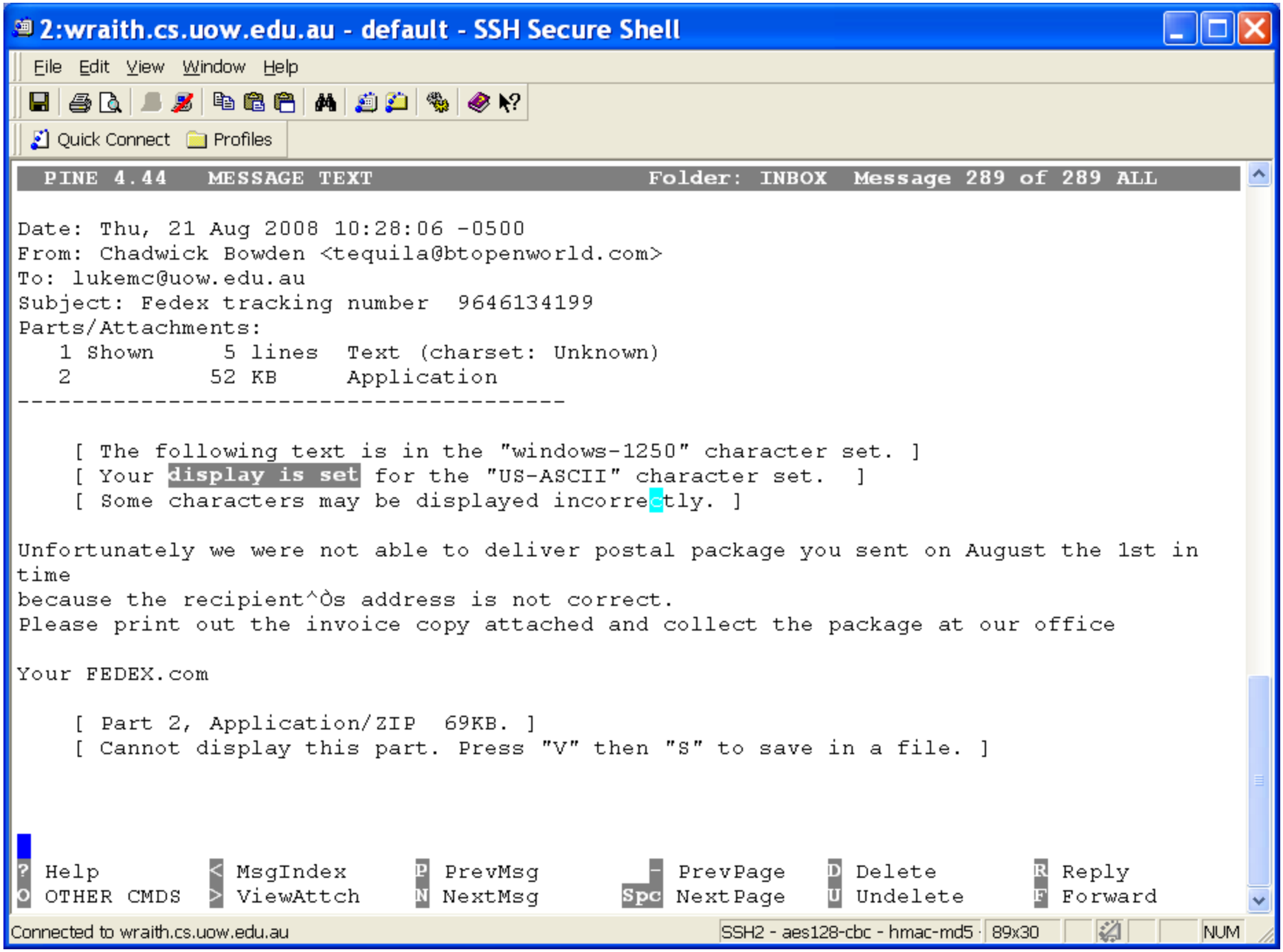


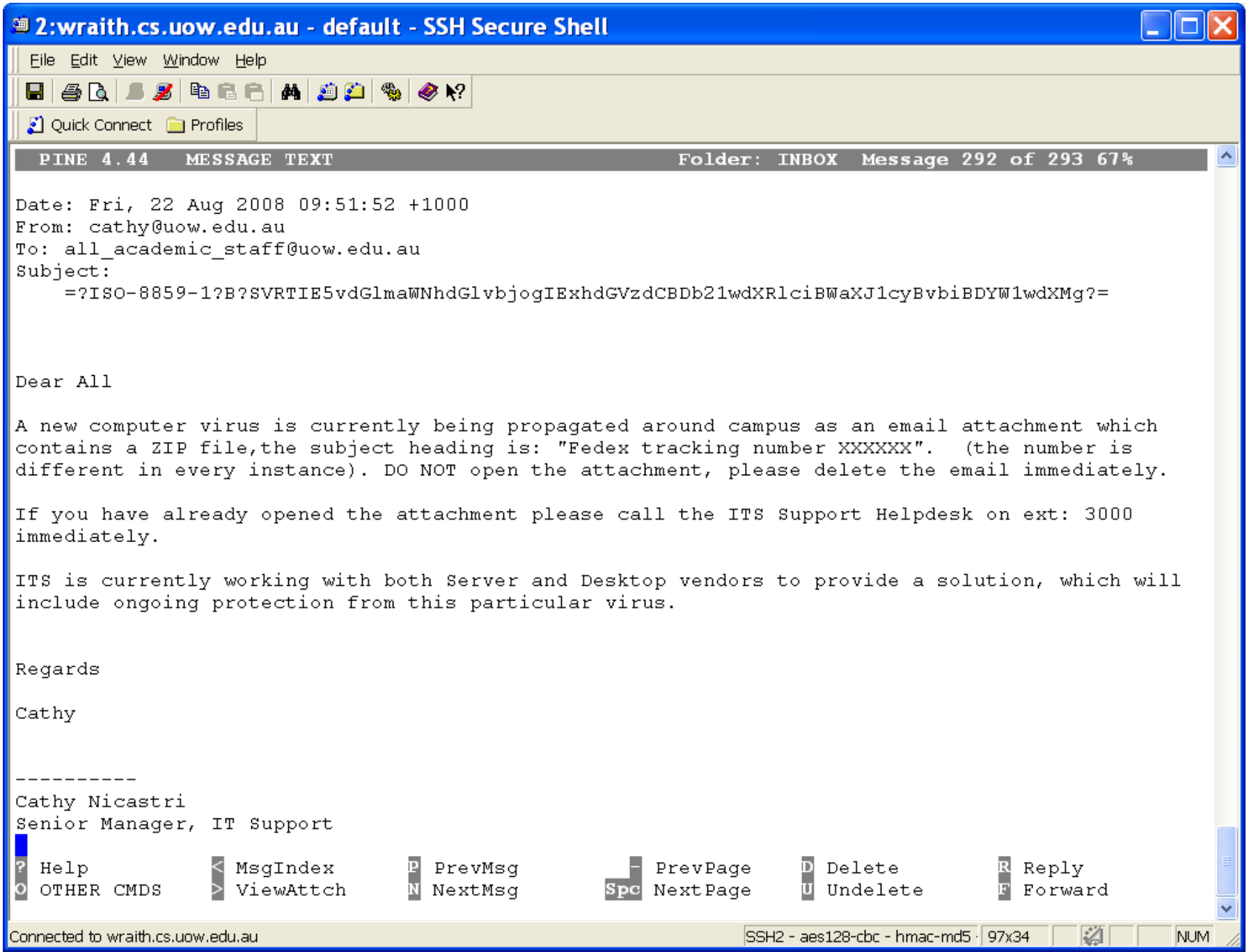
- This one uses some social engineering too …
- From https://en.wikipedia.org/wiki/Ransomware

Classification ...



From Stallings (Cryptography & Network Security)


2:wraith.cs.uow.edu.au - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
PINE 4.44 MESSAGE TEXT Folder: INBOX Message 289 of 289 ALL
Date: Thu, 21 Aug 2008 10:28:06 -0500
From: Chadwick Bowden <tequila@btopenworld.com>
To: lukemc@uow.edu.au
Subject: Fedex tracking number 9646134199
Parts/Attachments:
1 Shown 5 lines Text (charset: Unknown)
2 52 KB Application
[The following text is in the "windows-1250" character set.]
[Your display is set for the "US-ASCII" character set.]
[Some characters may be displayed incorrectly.]
Unfortunately we were not able to deliver postal package you sent on August the 1st in time
because the recipient^Òs address is not correct.
Please print out the invoice copy attached and collect the package at our office
Your FEDEX.com
[Part 2, Application/ZIP 69KB.]
[Cannot display this part. Press "V" then "S" to save in a file.]
? Help
O OTHER CMDS
< MsgIndex
> ViewAttch
P PrevMsg
N NextMsg
- PrevPage
Spc NextPage
D Delete
U Undelete
R Reply
F Forward
Connected to wraith.cs.uow.edu.au
SSH2 - aes128-cbc - hmac-md5
89x30
NUM


2:wraith.cs.uow.edu.au - default - SSH Secure Shell
File Edit View Window Help
Quick Connect Profiles
PINE 4.44 MESSAGE TEXT Folder: INBOX Message 292 of 293 67%
Date: Fri, 22 Aug 2008 09:51:52 +1000
From: cathy@uow.edu.au
To: all_academic_staff@uow.edu.au
Subject:
=?ISO-8859-1?B?SVRTIE5vdGlmaWNhdGlvbjogIExhdGVzdCBDb21wdXRlciBWaXJ1cyBvbiBDYW1wdXMg?=
Dear All
A new computer virus is currently being propagated around campus as an email attachment which contains a ZIP file,the subject heading is: "Fedex tracking number XXXXXX". (the number is different in every instance). DO NOT open the attachment, please delete the email immediately.
If you have already opened the attachment please call the ITS Support Helpdesk on ext: 3000 immediately.
ITS is currently working with both Server and Desktop vendors to provide a solution, which will include ongoing protection from this particular virus.
Regards
Cathy
Cathy Nicastri
Senior Manager, IT Support
? Help < MsgIndex P PrevMsg - PrevPage D Delete R Reply
O OTHER CMDS > ViewAttch N NextMsg Spc NextPage U Undelete F Forward
Connected to wraith.cs.uow.edu.au
SSH2 - aes128-cbc - hmac-md5 97x34
NUM

Viruses: “In the Wild”

- When a virus is in general circulation we refer to it as being "in the wild".
  - This implies computing environments outside of …
    - … the development environment where the virus was created and tested.
    - … the collections of antivirus vendors, researchers, and collectors.

“For a virus to be considered “In the Wild”, it must be spreading as a result of normal day-to-day operations on and between the computers of unsuspecting users.” Ducklin in 'Counting Viruses'

- The Wild List was started in 1993 by Joe Wells, with the associated organization founded in April 1996 by Joe Wells & Sarah Gordon.
  - http://www.wildlist.org
  - It includes an extensive list of currently Wild viruses.
  - Virus scanners should be able to detect all of the viruses in the Wild, as well as many older ones.

How Do Viruses Work?

- A virus is often quite a simple program, not just conceptually but also in implementation.
- A direct action virus can be modelled in terms of an algorithm such as the following:

```
begin
  Look for (one or more infectable objects)
    If (none found)
    then
      exit
    else (infect object or objects).
    endif
end
```

- Direct action viruses are only active when an infected object is active.

- A lot of viruses install themselves into the memory of the host computer when the original virus program is executed.
- This means that even after the original virus program is closed, new objects can be infected without having to run anything else.
- These are referred to as memory resident viruses.
- Hybrid viruses are both direct action and memory resident.

Viruses components

- We have seen that viruses have a structure allowing duplication.
- In addition to this **infection mechanism**, or infection vector or infective routine, there are two other components:
- A **Payload**, which is what, besides spreading the virus does.
- A **Trigger**, which is the condition to be met before the payload is activated.
- Most modern malware contains variants on each of these.

- The structure now becomes something like …

```
begin
  (Go resident)
  if (infectable object exists)
  then
    if (object is not already infected)
    then
    (infect object)
    endif
  endif
  if (trigger condition exists)
  then
    (deliver payload)
  endif
end
```

(Typical) Virus lifetime phases

- **Dormant phase**: The virus is idle. It may be waiting for a trigger before propagation begins.
- **Propagation phase**: The virus places a copy of itself into other programs or into certain system areas on the disk.
- **Triggering phase**: The virus is activated to perform the function.
- **Execution phase**: The function is performed.

Classification of Viruses

- Viruses can be classified according to the target …
  - Boot sector infectors (BSIs).
  - File infectors.
  - Macro viruses.
  - Scripting viruses.
- … and according to the method of concealment.
  - Encrypted.
  - Stealth.
  - Polymorphic.
  - Metamorphic.
- These are effectively technologies used by the virus.
- The transport mechanism is related to the target but can also be used as a means of classification.

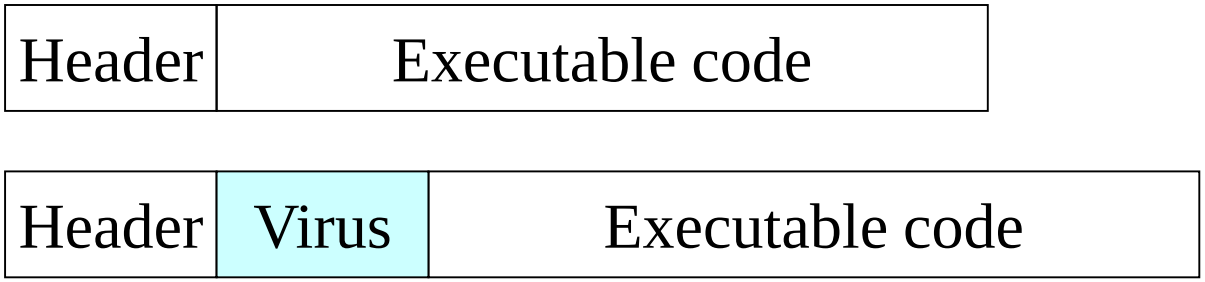
"The virus was contained in an e-mail warning about the virus . . ."

Boot sector infectors

- The boot sector is the part of a disk used to bootstrap the system or mount a disk.
  - It contains a Master Boot Record or a DOS Boot Record.
  - The Master Boot Record (MBR), or partition sector, is found only on hard disks, where it is always the first physical sector.
  - On floppy disks, which can't be partitioned and don't contain a MBR, the first physical sector is the boot record, or DBR (DOS Boot Record).
- Code in the boot sector is executed when the system sees the disk for the first time, and if the boot sector contains a virus the virus is executed.
- These viruses are less common now, since the use of floppy drives has almost disappeared.
  - USB malware is common though. ☺

File Infectors

- These types of virus insert the virus code into an executable, possibly .com or .exe on a PC, and that code runs when that program is executed.



Macro Viruses

- Viruses that are composed of a sequence of instructions that are interpreted, rather than executed directly.
- They can execute on any system that can interpret the instructions.
  - For example, a spreadsheet macro virus executes when the spreadsheet interprets these instructions.
  - The Melissa virus infected MS Word 97 & 98 documents on Windows and Macintosh systems.
  - They are not bound by machine architecture, rather by application structure.

- Macro viruses are generally easy to write.
  - Learning the syntax for a macro language might take a couple of days,
    - … against weeks for a high-level programming language like C++ (assuming you know a fair bit about programming).
    - … and against months for assembly.
- Windows OS is especially vulnerable, and because of the widespread use of, for example, Microsoft Office, they have the potential for a large target range.

Email Macro Viruses

- Visual Basic for Applications (VBA) was designed as a common macro language for use across a range of applications.
- VBA allows a virus writer to include, without great difficulty, statements querying a system to get all the necessary information (email application name, user's name and email password) and send an attachment via email.
- How did the Melissa virus read the address book of infected users to get 50 recipient's email addresses to send itself to?
  - The mailing itself used MAPI, or Messaging Application Programming Interface.

```
;Comments by Roger A. Grimes
Set UngaDasOutlook = CreateObject("Outlook.Application")
;creating an instance of Outlook
If System.PrivateProfileString("",
"HKEY_CURRENT_USER\Software\Microsoft\
Office\", "Melissa?") <> "... by Kwyjibo" Then
  If UngaDasOutlook = "Outlook" Then
;if Outlook is the email engine...
    DasMapiName.Logon "profile", "password"
;get email user's name and email password
    For y = 1 To DasMapiName.AddressLists.Count
;getting ready to count number of contacts in address book
        Set AddyBook = DasMapiName.AddressLists(y)
        x = 1
        Set BreakUmOffASlice = UngaDasOutlook.CreateItem(0)
        For oo = 1 To AddyBook.AddressEntries.Count
            Peep = AddyBook.AddressEntries(x)
            BreakUmOffASlice.Recipients.Add Peep
            x = x + 1
            If x > 50 Then oo =
                              AddyBook.AddressEntries.Count
         Next oo
;get up to 50 email addresses from address book
;end of Melissa code sample
```

Scripting Viruses

- These are based on script languages such as VBScript or Javascript.
- They can be thought of as Macro Viruses with respect to a scripting language.

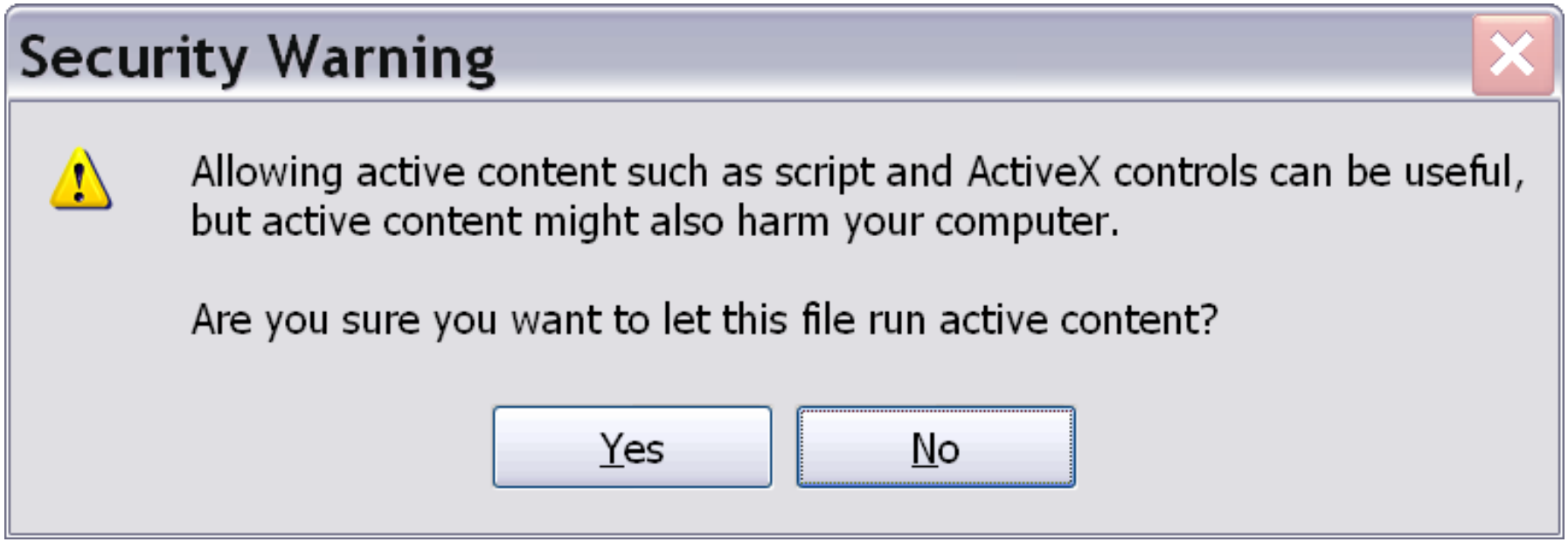
HTML.Internal

- One of the first HTML viruses, it was written for demonstration purposes.
- It will only work on browsers that handle VBScript and ActiveX.
  - It uses VBScript and calls to ActiveX to search for and infect HTML files.
- That effectively limits it to Internet Explorer, versions 4.0 and above.
- Default security should prevent the virus from spreading.

```
<html> <!--1nternal-->
<head>
<meta name="Author" content="1nternal">
<script language = "VBScript">
Sub Offline
       Set FSO = CreateObject("Scripting.FileSystemObject")
       HostPath = Replace(location.href, "file:///", "")
       cpath = fso.GetParentFolderName(HostPath)
       Set folder = fso.GetFolder(cpath)
       While folder.IsRootFolder = false
              Set folder = fso.GetFolder(cpath)
              Set fc = folder.Files
              cpath = fso.GetParentFolderName(cpath)
        WEND
End Sub
</script>
</head>
<BODY onLoad="Offline">

</BODY>
</HTML>
```

What is happening?

- HTML.Internal checks to see if the URL includes *file:*
  - If it does it implies the browser is operating in offline mode with a file being viewed on the local hard drive.
  - If allowed to successfully run, this virus will attempt to infect all *.HTML* files in the local and parent directories.
  - The infectious code is inserted into the host *.HTML* file near the beginning and called when the file is launched (with the *OnLoad* event).
- As we have seen in some examples of unsafe HTML documents, the browser will generally warn you that an unsafe ActiveX control is trying to execute, and it will prompt you to accept or deny the launch.


Security Warning
Allowing active content such as script and ActiveX controls can be useful, but active content might also harm your computer.
Are you sure you want to let this file run active content?
Yes
No


Internet Explorer
An ActiveX control on this page might be unsafe to interact with other parts of the page. Do you want to allow this interaction?
Yes
No

Some variations…

- Look for the FileSystemObject in Scripts.
- Find a file in a local disk.
- Insert something to the file.
- How dangerous is it?
  - It could erase the hard disk.
  - It could delete files.
  - ...

```
<html> <!--1nternal-->
<head>
<meta name="Author" content="1nternal">
<script language = "VBScript">
Sub Offline
       Dim fsysobj, foldr, file1, s
       Set fsysobj =
              CreateObject("Scripting.FileSystemObject")
       Set file1=fsysobj.GetFile("C:\AUTOEXEC.BAT")
       document.write(file1.size)
       Set ts = file1.OpenAsTextStream(8, -2)
       ts.write "Format the disk"
       ts.close
End Sub
</script>
</head>
<BODY onLoad="Offline">
</BODY>
</HTML>
```

8: Open, and write to the end.

-2: System default, ASCII/Unicode.

Create a directory

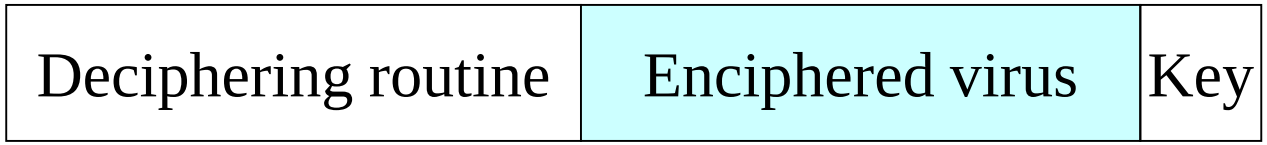
```
<html>
<head>
<meta name="Author" content="1nternal">
<script language = "VBScript">
Sub Offline
       Dim fsysobj, foldr
       Set fsysobj =
            CreateObject("Scripting.FileSystemObject")
       Set foldr = fsysobj.CreateFolder("C:\Malicious")
End Sub
</script>
</head>
<BODY onLoad="Offline">

</BODY>
</HTML>
```

Execute a program

```
<HTML>
<body>
<SCRIPT language="JavaScript">
objectx= new ActiveXObject ("WScript.Shell");
string="command /k echo It Worked!";
objectx.Run(string);
string="command /k edit";
objectx.Run(string);
</script>
</body>
</HTML>
```

Encrypted Viruses

- Viruses which are encrypted with a cipher.
- Why?
  - To avoid detection. The virus code is hidden!
- How?
  - The virus code is encrypted, except for the decryption routine and key.
- Does it work?
  - We will look at detection a little later.



An Example of an Encrypted Virus

- From the book: “A Pathology of Computer Viruses” by Ferbrache.

```
// initialize the registers with the keys
rA <- k1;
rB <- k2;
// initialize the rC with the message
rC <- sov;
// the encryption loop
while (rC != eov) do begin
      // encrypt the byte of message
      *rC <- *rC xor rA xor rB
      rC <- rC + 1;
      rA <- rA + 1;
end
```

Stealth viruses

- Viruses with this technology explicitly try to hide all of themselves from detection.
- One typical approach is to include compression.
  - Detecting that a file has changed by checking the length won't work anymore.

```
    program CV :=

{goto main;
    01234567;

    subroutine infect-executable :=
          {loop:
               file := get-random-executable-file;
          if (first-line-of-file = 01234567) then goto loop;
        (1)     compress file;
        (2)     prepend CV to file;
        }

main:   main-program :=
          {if ask-permission then infect-executable;
        (3)     uncompress rest-of-file;
        (4)     run uncompressed file;}
        }
```

Polymorphic Viruses

- Polymorphic viruses change form each time they are inserted into another program.
- They change the instructions in the virus to something equivalent but different.
- If the viruses is encrypted, as is fairly common, the decryption code is the segment of the virus that is changed.
  - For example, at the instruction level, all the following have exactly the same effect.

```
add 0 to operand
Logical AND 1 with operand
no operation
subtract 0 from operand
```

Example of Polymorphic Viruses

```
// initialize the registers with the keys
rA <- k1;
rD <- rD + 1; // random line
rB <- k2;
// initialize rC with the message
rC <- sov;
rC <- rC + 1; // random line
// the encryption loop
while (rC != eov) do begin
    rC <- rC – 1; //random line x
    //encrypt the message
    *rC <- *rC xor rA xor rB;
    //advance all the counters
    rC <- rC + 2; //counter incremented
    //to handle random line
    rD <- rD + 1; //random line
    rA <- rA + 1;
end
while (rC != sov) do begin //random line
    rD <- rD – 1; //random line
end //random line
```

The lines marked random have no effect. This is equivalent to the earlier code.

Metamorphic viruses

- These are, in some sense, a higher order of polymorphic viruses.
- Not only do they change in form between transitions they can be completely re-written.
  - They can re-write in a version suitable for executables on a different platform too.

Trojan horses

- A friend passes you an interesting game.
  - You run it and it is lots of fun.
  - The license says it's freeware so you pass it to your friends.
- What could happen?
  - It could be a Trojan Horse.
  - It could be doing something malicious in addition to the obvious game.
- Computer Trojan Horses are programs with overt (documented or known) effects and covert (undocumented or unexpected) effect.

Trojan Technology

- Some concealment methods:
  - A Trojan renames itself to the name of a valid system file.
  - They can also be encrypted and polymorphic, and could install themselves in different ways to escape detection.

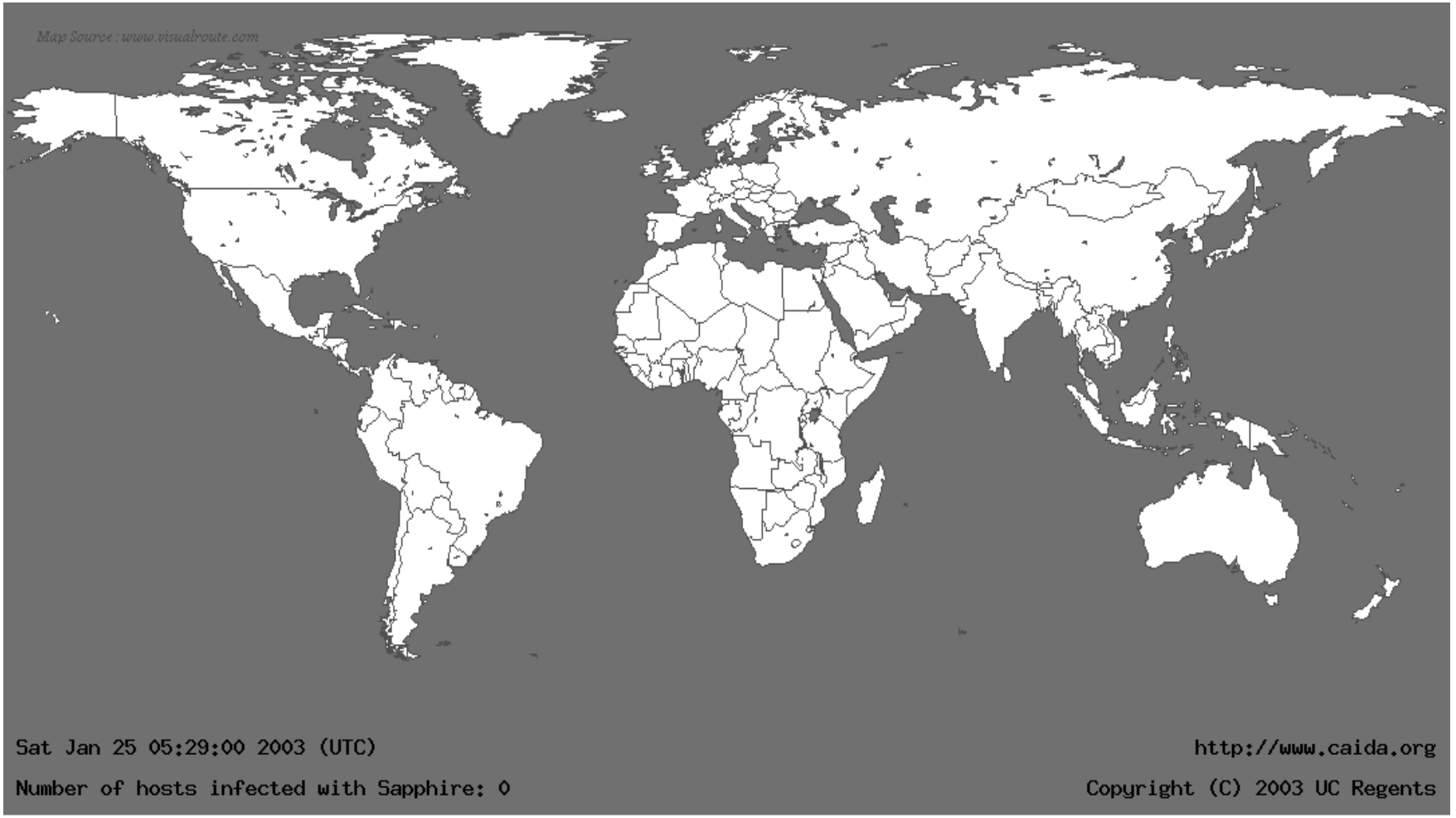
- Hiding as source code:
  - They transfer themselves as ASCII text source code onto the host machine.
  - They are then compiled or interpreted to bypass malicious code scanners.
  - The executable code or a batch file is included to assemble or interpret the code on the fly.
  - The companion programs that assemble or link the source code into its runtime form are usually legitimate programs and will not be flagged by scanners either.
  - Other Trojans use tools already available on most Windows PCs (*DEBUG.EXE or WSCRIPT.EXE*) to launch their programs.

- Remote Administration Trojans (RAT's):
  - These allow a hacker to take complete control of a PC. Very nasty with keyboard & screen capture, and the ability to directly manipulate your computer.
- Backdoor Programs:
  - In March 2001 it was revealed that a group of Eastern European hackers had spent over a year exploiting an NT vulnerability and installing backdoor Trojans to steal more than a million credit cards from over 40 top e-commerce and e-banking web sites.
- Network Redirection:
  - This allows a hacker to redirect specific attacks through a compromised intermediate host machine toward a new target.

Worms

- A program that copies itself from one computer to another.
- Host computer worms are entirely contained in the computer they run on and use network connections only to copy themselves to other computers.
- The original may terminate itself after launching a copy on another host - sometimes called "rabbits."

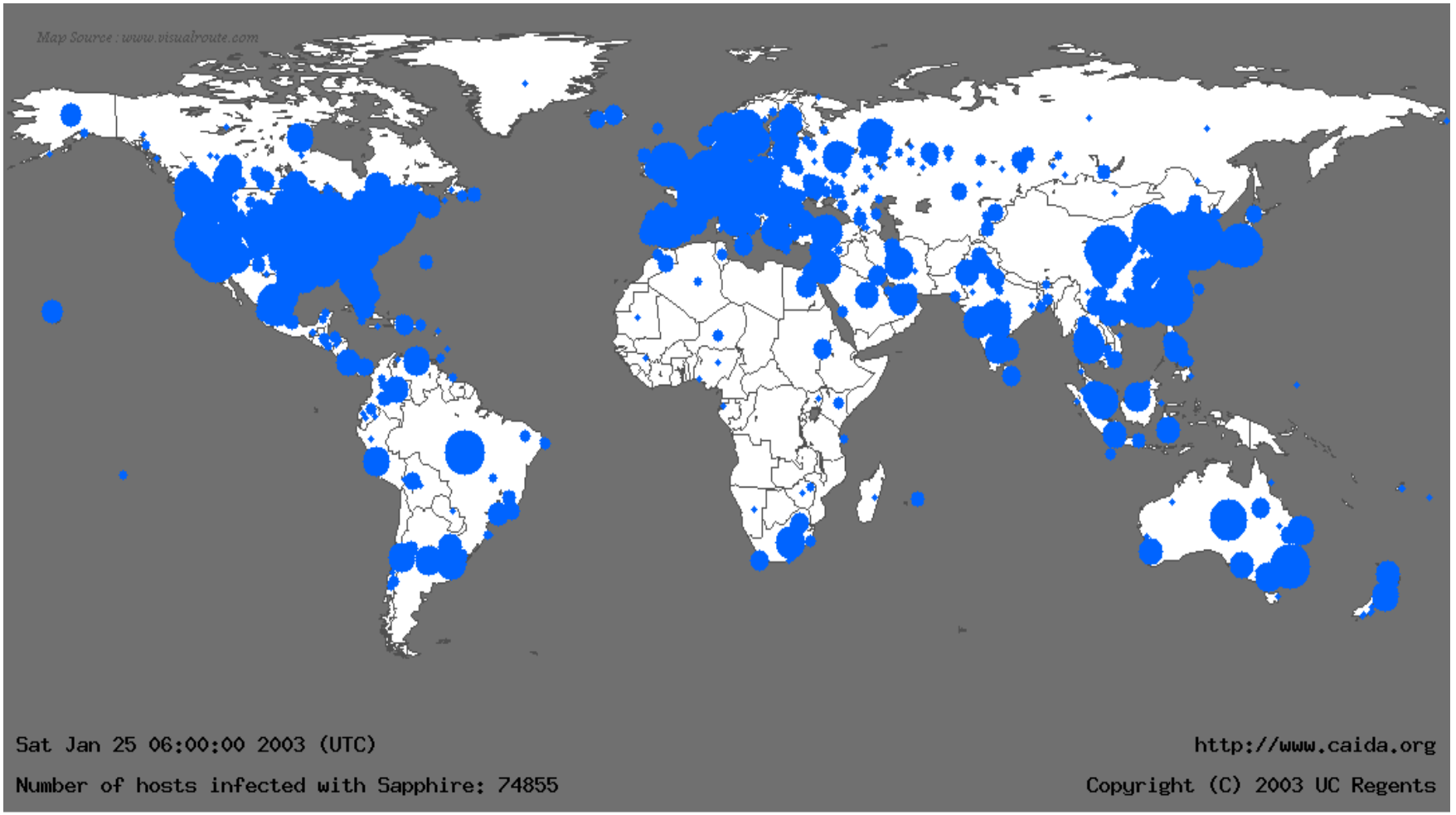
The world: Pre-Slammer Worm



From www.caida.org: Lots of interesting things.

Exploits buffer overflow in Microsoft SQL Server 2000.

31 minutes later ☹



Look at 6a_Code-Red.wmv too

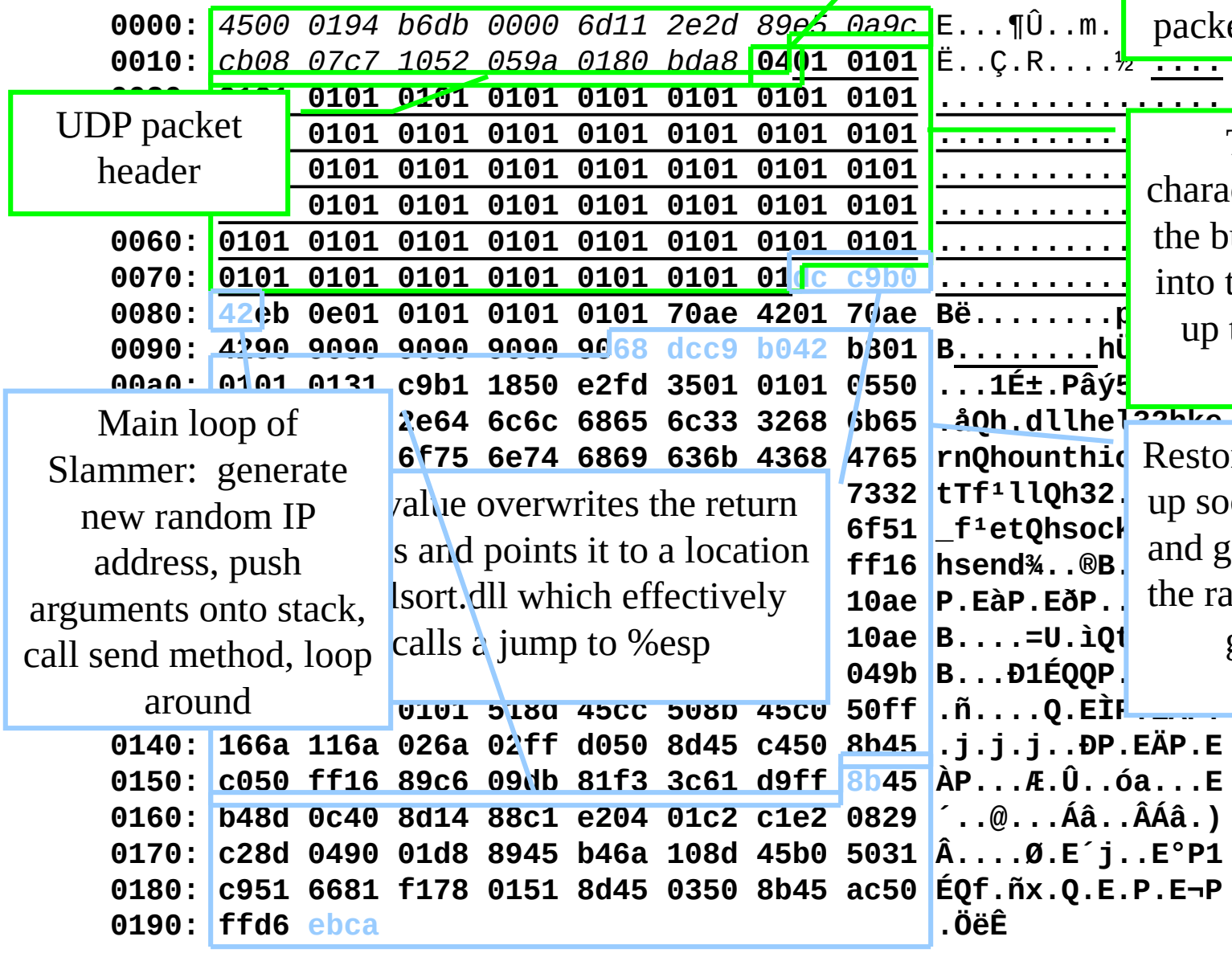
Why was Slammer so fast?

- 90% of the vulnerable hosts were infected within 10 minutes.
- The spread doubled every 8.5 seconds and the highest scanning rate of 55 million scans per second was reached in 3 minutes.
- Slammer used UDP, which is connectionless, unlike TCP which involves responses.
- As such it wasn’t limited by the latency in communicating back to the source host, it was limited just by the bandwidth from an attack station and the speed of transport to other hosts.

The broadcast mechanism

- The Slammer code randomly generated an IP address and sent out a copy of itself to that location, using UDP.
  - Slammer exploited a vulnerability in MS SQL Server Resolution Service, for which a patch had been available for 6 months.
  - If a UDP packet arrived on port 1434 with the first byte 0x04, the rest of the packet is interpreted as a registry key to be opened, and is stored in the buffer for that purpose later.
  - But without length checking the rest of the packet can be written into the buffer.
- The packet was just 376 bytes long.

SQL Slammer Worm UDP packet



This byte signals the SQL Server to store the contents of the packet in the buffer

UDP packet header

The 0x01 characters overflow the buffer and spill into the stack right up to the return address

Main loop of Slammer: generate new random IP address, push arguments onto stack, call send method, loop around

value overwrites the return s and points it to a location lsort.dll which effectively calls a jump to %esp

Restore payload, set up socket structure, and get the seed for the random number generator

Bacteria

- A bacterium is a program that creates many instances of itself to burn up resources of some type.
  - A bacterium is not required to use all resources on the system, but is aiming to result in some level of denial of service.
- Some examples …

```
while true
do
    mkdir x
    chdir x
done
```

Logic Bombs

- A program that performs an malicious action when some external event occurs.
- Plant Trojan horses in system using a logic bomb.
- The most common logic used is date matching, but it could be anything.

Defence

- Malware acts as both data and instructions:
  - A virus inserts code (a set of instructions) into another program.
  - The set of instructions is treated as data.
  - The virus executes itself, the set of instructions is treated as an executable.
  - Protection:
    - Treat all programs as type "data“.
    - Some certifying authority can change the type to executable, after verification takes place.

- Against Malicious code assuming the identity of a user:
  - When a user executes malicious code, that code can access and affect objects within the user's protection domain.
  - Protection: Limiting the objects accessible to a given process run by the user.
    - Information Flow Metrics.
    - Reducing the Rights.
    - Sandboxing.

- **Information flow metrics**:
- Define the flow distance metric *fd*(*x*) for some information x as follows:
  - Initially, all information has *fd*(*x*) = 0.
  - Whenever *x* is shared, *fd*(*x*) increases by 1.
  - Whenever *x* is used as input to a computation, the flow distance of the output is the maximum of the flow distances of the input.
- Information is accessible only while the distance is **less** than some value V.

- Example of applying the flow distance metric.
  - Users A, B, and C work on the same computer.
  - V_A =3. V_B=V_C=2.
  - A creates a program P containing a virus. ☺
  - B executes P.
    - The contents of P have a flow distance of 0, so when the virus infects B's file Q, the flow distance of the virus is 1, and so B can access it.
      - Hence, the copying succeeds.
  - C executes Q, when the virus tries to spread to her files, its flow distance increases to 2.
    - Hence, the infection is not permitted, because C can only access information with a flow distance 0 or 1.
    - C can however execute P and it will flow. ☹

- **Reducing the Rights**:
  - The user can reduce their associated work domain when running a suspect program.
  - This follows from the principle of least privilege:

> A subject should be given only those privileges that it needs in order to complete its task.

More on defence

- Sandboxing can be used.
  - Suspecious codes are quarantined in an isolated system area
  - Then run the code and monitor its behavior.
- Restrict sharing by controlling the domain boundaries:
  - Restrict users in different protecting domains from sharing programs or data.

Detection

- Normal behaviour of a system is usually different from the activity profile of an infected system.
  - Virus monitors monitor known methods of virus activity, such as attempts to write to a boot sector, modify interrupt vectors, write to system files... and detect abnormal behaviour of the system.
- Advantages:
  - Works for all viruses.
  - Detection is before (complete) infection.
- Disadvantages:
  - To detect a high percentage of viruses, the sensitivity of the monitor must be set high and this may generate many false alarms.

- Theorem (by Cohen):

  It is undecidable whether an arbitrary program contains a computer virus.

There are formal definitions of viruses that allow this type of result to be derived.

Multiple copy testing …

- Run several copies of the “same program or algorithm”.
  - This is not simply running the same program several times.
- Majority rules…
- The check could be based on results, calls made, efficiency …
- Majority still might not be trusted, because they might all be corrupted.
- But different performances can imply action needs to be taken.

Signature scanning

- Signature scanning (signature= search string= scan string) is the simplest and the most common approach to virus detection.
- Signature extraction is a non-trivial process:
  - The infection is disassembled and the key portions are identified.
  - The key portions are combined to form a signature.
  - The signature is checked against a large library of programs to reduce the chance of false positives occurring when signature accidentally matches some library code.

- Advantages:
  - Signature scanning can be used against Trojan horses, logic bombs and other malicious software.
- Disadvantage:
  - Scanning cannot find new viruses before their patterns are known.
  - It is also ineffective against polymorphic viruses.
- First-generation scanner's use virus signatures only.
- GPU's can be used to speed up signature processing!

http://http.developer.nvidia.com/GPUGems3/gpugems3_ch35.html

2nd, 3rd and 4th generation scanners

- Second generation scanners don't just use specific signatures, since the signatures of many viruses change (polymorphic).
- It uses a lot of optimised search mechanisms, e.g. looking for code that are often associated with malware (encryption, compression, etc.)

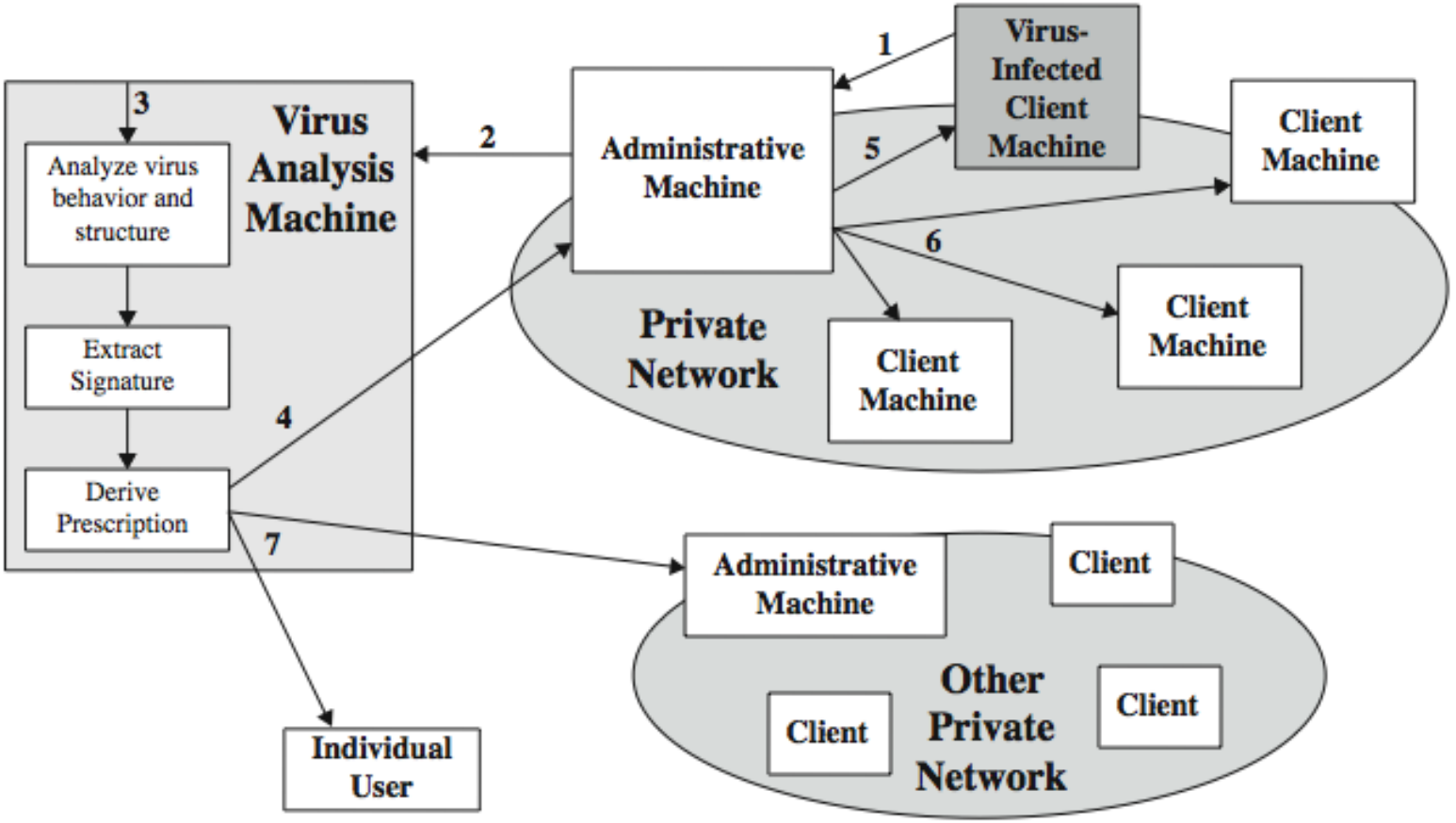
- Or they might use integrity checks…

- … Manipulation Detection Codes (or MDC's).
  - Apply some function to a file to obtain a set of bits called the signature block and then protect that block.
  - If the recomputing the signature block, the result differs from the stored signature block, the file has changed, possibly as a result of a malicious code.
  - We can use keyed hash functions here!

- … and timestamps are useful too.
- The time of the last change to a program is kept separate from the environment that the program is stored.
- Timestamps should be frequently checked to ascertain the integrity of the program.

- Third generation scanners detect viruses by behaviour.
  - For example, attempts to interact inappropriately with certain system files could trigger detection.
- Fourth generation basically use a collect of antivirus techniques together.

Digital Immune System



Originally IBM, subsequently Symantec.

Figure 6.6, page 213 of Stallings and Brown.

- Objective: To provide a rapid response so viruses can be stamped out soon after being introduced.
  - On detecting a new virus, the immune system captures and analyses it, adds detection and shielding information, removes it, and passes information about that virus to other systems so it can be detected before being allowed to run elsewhere.

1. Monitoring programs on the PC's use heuristics to infer a virus may be present. A copy is passed to an administrative machine.

2. Admin machine encrypts the "virus" and sends it to a central virus analysis machine.

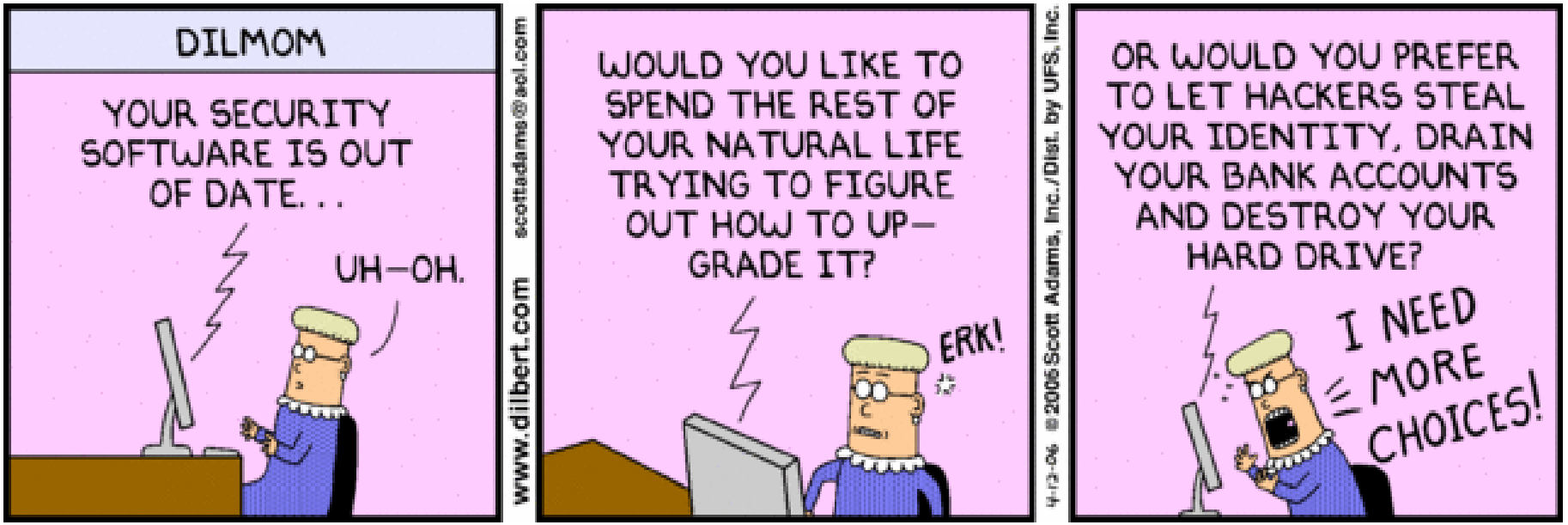
3. The central virus analysis machine provides a safe environment for analysis, and produces a prescription for virus identification and removal.

4. Prescription sent back to the admin machine.

5. Prescription forwarded to the infected client.

6. Prescription forwarded to other clients in the organization.

7. Worldwide subscribers get regular antivirus updates.



Dilbert: 12-Apr-2006

CSCI262
System Security (S7a)

Intrusion detection Systems (IDS)

Overview

- What is intrusion detection?
  - Categories of attacker.
- False positives and false negatives (again).
- Models: Anomaly, Signature/Misuse.
- Architecture and organisation.
- Intrusion response.
- SNORT.
- “Active intrusion detection systems”.
  - Honeypots.
- Intrusion prevention systems.

What is intrusion detection?

- Intrusion detection is detecting the *circumvention of a policy*.
  - In particular security policies but we don't necessarily need to separate them out in particular.
  - This means we failed to enforce a policy → ☹
- Intrusion detection is closely related to system auditing, and underlying both we need to record what is going on in a system.

- Intrusion detection systems monitor the behaviour within a system with the mindset that there may have been intrusions.
  - That is, the mechanisms for enforcing our policies might not work all the time.
  - Remember access control related to both policies and mechanisms.
- We need to distinguish between intrusions and legitimate behaviour.
- Some types of intrusions are significantly more visible than others.

Categories of attacker

- Attackers roughly fall into 1 of 3 classes.
- **Clandestine**: These try to avoid the intrusion detection or auditing system.
- **Masqueraders**: These pretend to be a legitimate user.
  - So if Oscar has guessed Alice's password, Oscar can masquerade as Alice.
- **Misfeasors**: These are legitimate users who are misusing the privileges they have.
  - These can often be difficult to determine...

Masquerader vs misfeasor vs legitimate

- All have the password for a legitimate account.
- How can we distinguish between them?
- Maybe on the basis on where or when they log in.
- Likely primarily on the basis of what they do once they are logged in.

A note on IDS vs IPS

- Intrusion detection can be “extended into prevention.”
  - This active extension is then an Intrusion Prevention System (IPS) or Intrusion Detection and Prevention System (IDPS).
- So, for example, an IPS may detect port scanning, which is allowed behaviour, but may be indicative of attack under way, so may actively work to block the attack.

- From Whitman & Mattord, Principles of Information Security, referring to the NIST 800-94 guide:
- *“IPS technologies are differentiated from IDS technologies by one characteristic: IPS technologies can respond to a detected threat by attempting to prevent it from succeeding.”*

“Normal system behaviour”

- Denning (1987) described three characteristics that allow us to determine systems that are behaving normally, or ones that are not.

1) Interactions between subjects and objects follow statistics.
2) Sequences that are likely to circumvent the policies would not be part of typical sequences of behaviour.
3) Processes have a specification set, or a set of allowed actions.

Models

- Behaviour inconsistent with one or several of those characteristics may be taken as evidence of an attempt of an attack.
- A model or strategy allows us to characterise a sequence of states or actions.

- We have two basic strategies, and we will look at each briefly.
- The first is **anomaly detection**, for when observed behaviour differs from the typical behaviour for a user.
    - This requires statistics on typical user behaviour, for individual users.

- The second is **signature or misuse based**, when observed behaviour indicates an attempt to inappropriately use resources.
  - This requires that we know typical attack patterns.
- In essence, anomaly approaches attempt to define normal, or expected, behavior, whereas signature-based approaches attempt to define improper behavior.

Goals of intrusion detection

- Before we look at strategies in more detail it's useful to identify the goals of intrusion detection systems, and to talk about accuracy.

1) **Detect** all intrusions, internal and external

2) **Dynamic** in the sense of capable of learning or taking into account current attacks and/or user behaviour.

3) **Timely**, in the sense of providing information at a point in time at which it is still useful.
4) **Clear and Concise** Reports of the results of the analysis.
5) **Accurate**. A incorrect report isn’t good, but a correct report that we don’t believe isn’t good either.

False positives and negatives…

- Hopefully you can all remember these terms from authentication, in the context of attempting to distinguish between authorised and unauthorised entities.
  - They are to do with the likelihood of getting a result which is wrong.
  - The interpretation in the context of intrusion detection systems is a little different from that of authentication though.

- **A false positive** is when make a match but “shouldn’t have”.
  - In an intrusion detection system we will be aware of a false positive, which we wouldn’t have been with authentication in the context of a false positive.
- **A false negative** is when we don’t make a match but “should have”.
  - This time it’s the intrusion detection system where we are unaware, and the authentication system where we are aware.

Anomaly Intrusion detection

- We are going to illustrate this in terms of the user behaviour aspect
- We need a way to characterise user behaviour, in a measurable way.
- We can look at measurements associated with individual activities, or with many activities, and we can also look at the actual mix of activities too.

- Typically the measurables for individual activities can be characterised by time density and event length distributions.
  - How many at a time? And each for how long?

- Measures might include CPU usage, number of processes, number of files opened, typically length of file open time, or process run.

- Depending on the user some of the characteristics may be more significant than others, even within the same system.
- In terms of activity mixes this might be something distinguishing between something like:

edit file, close file, compile file, run process, edit file, close file, compile file, run … **and**

edit file, edit file, edit file, edit file, edit file, edit file, edit file, …

Statistics: A threshold model

- The simplest statistical model for modelling anomalies is a threshold model.
  - If more than a certain number of something happens an alarm is triggered.

  Or

  - If less than a certain number of something happens an alarm is triggered.
- Login attempts for example.

Mean and standard deviation

- The threshold can be turned into slightly more than a count, particularly if we are dealing with a continuous measure rather than a discrete one.
- Effectively we can specify an anomaly as being a value more than a certain number of standard deviations from the mean.

Dilbert: 05-Dec-2006

Table 8.2 Measures That May Be Used for Intrusion Detection

Measure	Model	Type of Intrusion Detected
Login and Session Activity		
Login frequency by day and time	Mean and standard deviation	Intruders may be likely to log in during off hours.
Frequency of login at different locations	Mean and standard deviation	Intruders may log in from a location that a particular user rarely or never uses.
Time since last login	Operational	Break-in on a "dead" account.
Elapsed time per session	Mean and standard deviation	Significant deviations might indicate masquerader.
Quantity of output to location	Mean and standard deviation	Excessive amounts of data transmitted to remote locations could signify leakage of sensitive data.
Session resource utilization	Mean and standard deviation	Unusual processor or I/O levels could signal an intruder.
Password failures at login	Operational	Attempted break-in by password guessing.
Failures to login from specified terminals	Operational	Attempted break-in.

Operational: based on a judgment of what is considered abnormal, rather than an automated analysis of past audit records.

Multiple variables: Measures in more detail …

- A collection of measures, (m_1, m_2, …, m_n) and associated weights (w_1, w_2, … w_n).
- Each user has an appropriately determined base profile (M_1, M_2, … M_n).
- To process a system we measure the active profile (μ_1, μ_2, … μ_n).
- We then apply a collection of distance functions D_i to determine $d_i=D_i(M_i,\mu_i)$.

- Our decision can then be based on a threshold, either at the level of individual distances, or in terms of a composite threshold

$$\sum_{i=1}^{n} w_i d_i \leq d_t$$

- It could also be possible to cross-correlate the distance functions.
- The distance functions themselves are functions of the probability distributions associated with the user profile and their actual profile, that is the result of observations.

Aging of data ...

- We shouldn’t really rely heavily on old statistics.
- If we are accumulating data over a significant period of time, and taking it all into account, we should weight the data as a function of time.

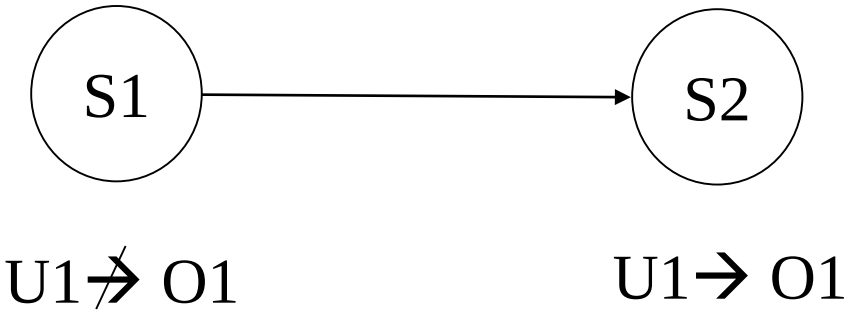
Signature based Intrusion detection

- These look for specifically sequences of events, or resource usages, or some complex condition sets, that describe a known attack.
- Such patterns are sometimes referred to as intrusion signatures.
- Once an attack has been attempted a signature can be extracted and used to detect later instances of the same attack.

- Misuse modelling usually refers to misuse from *within* the system by an authorised user.
- If a set of actions by a user matches to a set of known rules, an intrusion is reported.
- Expert systems are often used to analyse the data.
- Some examples are:
  - Intrusion Detection In Our Time (IDIOT).
    - Monitors Audit logs and looks for sequences of events that correspond to an attack.
  - STAT.
  - Network Flight Recorder (NFR).

STAT

- This monitors the security state of the system.
- If it goes from a less privileged state to a more privileged state, the way this transition has occurred is monitored.

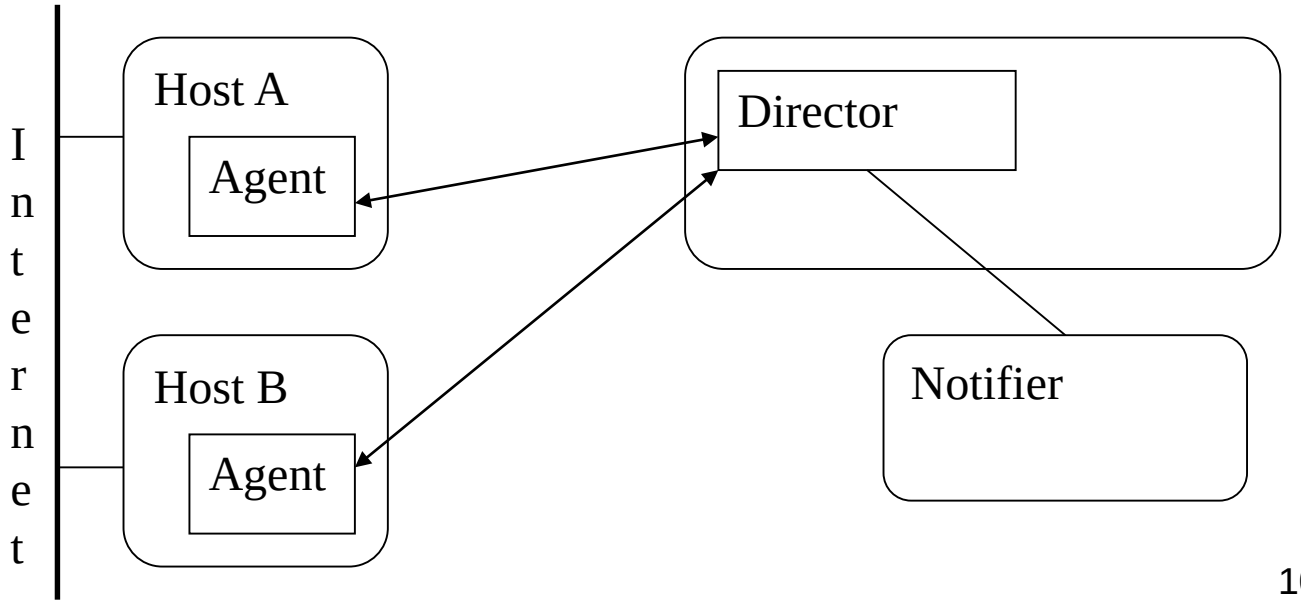


Network Flight Recorder (NFR)

- This intrusion detect tool takes packets from the network and filters them.
- A backend writes the information generated by these filters to disk.
- Administrators can then query this backend without impacting network performance.

Architecture of an IDS/IDPS

- The architecture involves:
  - Agents – Gather information from loggers/sensors and likely perform some analysis.
  - A Director – Gather information from agents and perform some analysis.
  - Notifier.



Agents

- An agent collects data from sources, including log files, a network or other processes.
- The agents pre-process information before it gets to the director.
  - For example, agents may transfer failed login attempts from a log, and discard successful logins.
- The director can, however, request more information from the agents.
  - This is particularly important where the correlation between events across agent domains is significant.

How do agents gather information?

- Host-Based information gathering (HIDS/HIDPS):
  - Looks primarily at log files, in the context of a host and likely particular applications.
  - Analyse them to determine what to pass to the director.
    - The events to look for are those relating to the goals of the IDS.
- Network-Based information gathering (NIDS/NIDPS):
  - Monitor network traffic
  - It is difficult if network traffic is encrypted. ☹

Capturing traffic: SPAN

- Facilitate “passive access to network traffic”.
  - They are used in IDS/IPS and sniffers.
- Switched Port Analyzers (SPAN):
  - Switches are used to support traffic flow, typically in the bottom 4 layers.
  - Traffic onto particular ports is copied or mirrored to an analysis dedicated port.

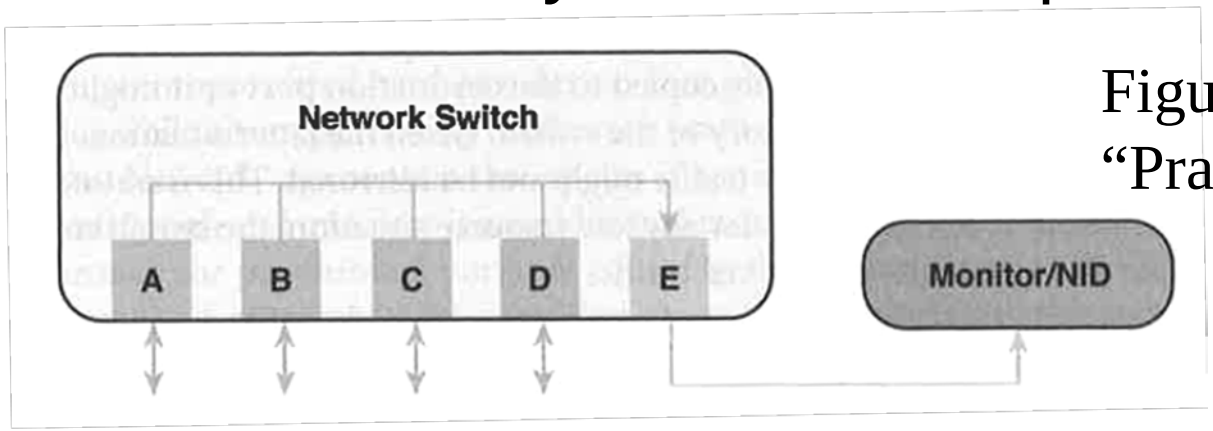


Figure 2.2 From Trost
“Practical Intrusion Analysis”

- Combining sources:
  - In Unix, an application level log will be significantly different to a system level log.
  - From the system level view, application logs are insufficient.
  - From the application level view, system logs are insufficient.
  - An agent, or the director, determines the appropriate level, or maps the information to that level.
- One advantage of combining information from multiple sources is the possibility of anomaly detection, used in anomaly based IDS.

- Consider this example of correlating information from multiple logs:
  a. Alice normally logs in during the day to perform system maintenance.
  b. Alice occasionally logs in during the late evening to write reports.
  c. One day, Alice logs in the late evening and changes the system kernel.

  The agent provides logs of both login times, and the commands executed separately.

  All appear normal, but looking at them all together we see anomalous behaviour.

Why use Agents?

- Traditional IDS have one point of failure, the director.
- To overcome this we need an IDS where multiple components would function independently, yet still be able to correlate information.
- If one agent has been attacked, the others would still continue to monitor the network.
- What is the cost?
  - There will be an overhead in the communication.

Director

- The director further analyses information, using an analysis engine (Intrusion Detection Expert System).
- The director can instruct the agents to:
  - Collect or send more information.
  - Process data differently.
- The director usually requests more information when it detects an attack.
- An agent can obtain information from a set of hosts, in this case it can act as a director with respect to those hosts.
- A director usually runs on a different system, so attackers can’t compromise it at the same time as the system we are trying to protect.

Notifier

- Notifies the appropriate party regarding reports received from the director.
- The notifier can be responsible for coordinating IDPS residing on firewalls to block attacks over the network.
- If an attack is identified the notifier can instruct other IDPS's to counteract the attack.

Combining Host and Network Monitoring

- The Distributed Intrusion Detection System (DIDS) was one of the early models, developed in the late 1980's, early 1990's by the US Department of Defence.
  - It is an example of a combined HIDS-NIDS.
- Host only or network only monitoring tends to be ineffective.
  - Logging into a system without a password wouldn't be detected by network monitoring.
  - However, subsequent actions may be detected by host monitoring.

- A DIDS director monitors agents that are attached to hosts as wells as monitoring network traffic.
- Agents scan logs for events of interest and report them to the DIDS director.
- The DIDS director performs analysis using an expert system.
- Note that combining HIDS and NIDS pretty much implies a distributed system.

DIDS

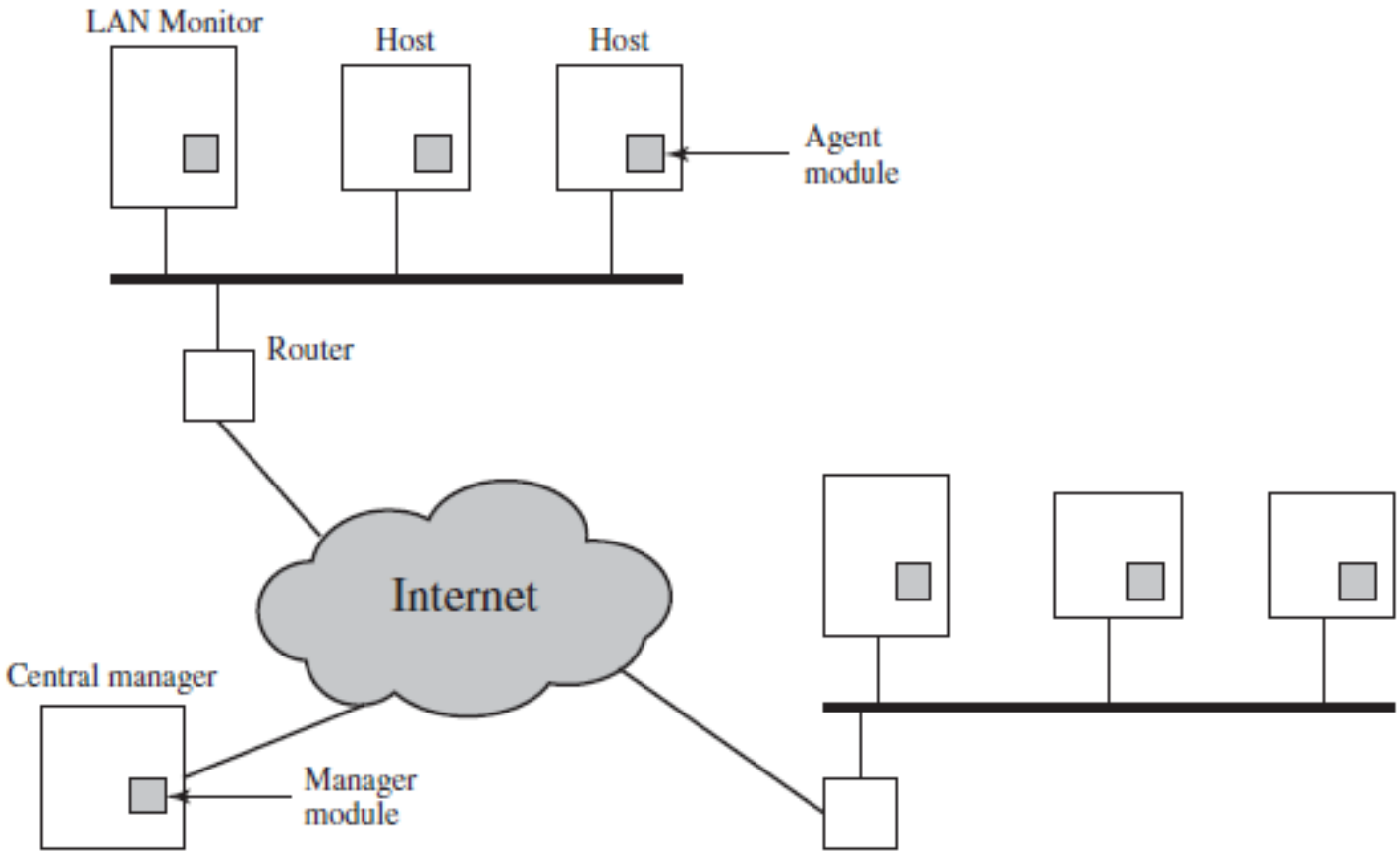


Figure 8.2 **Architecture for Distributed Intrusion Detection**

Intrusion Response

- Incident Prevention and Handling:
  - If we can we would like to try and prevent the intrusion from succeeding!
  - But we may not be able to, and appropriate handling requires that various mechanisms be in place:
    1. Preparation for an attack.
    2. Identification of an attack.
    3. Containment of the attack.
    4. Eradication of the attack.
    5. Recovery from the attack.
    6. Follow-up to the attack.

Look at …

- The site security handbook, RFC 2196.
- In particular the material towards the end.

- www.ietf.org
- All internet related technologies and standards

- **Preparation for an attack**:
  - Procedures and mechanisms for detecting and responding to attack must be set up before any attacks are detected.
  - Effectively you need to know in advance what to do if an attack is identified.
  - Why do we have fire drills?
- **Identification of an attack**:
  - When an attack is identified the rest of the phases come into play.

Containment of the attack

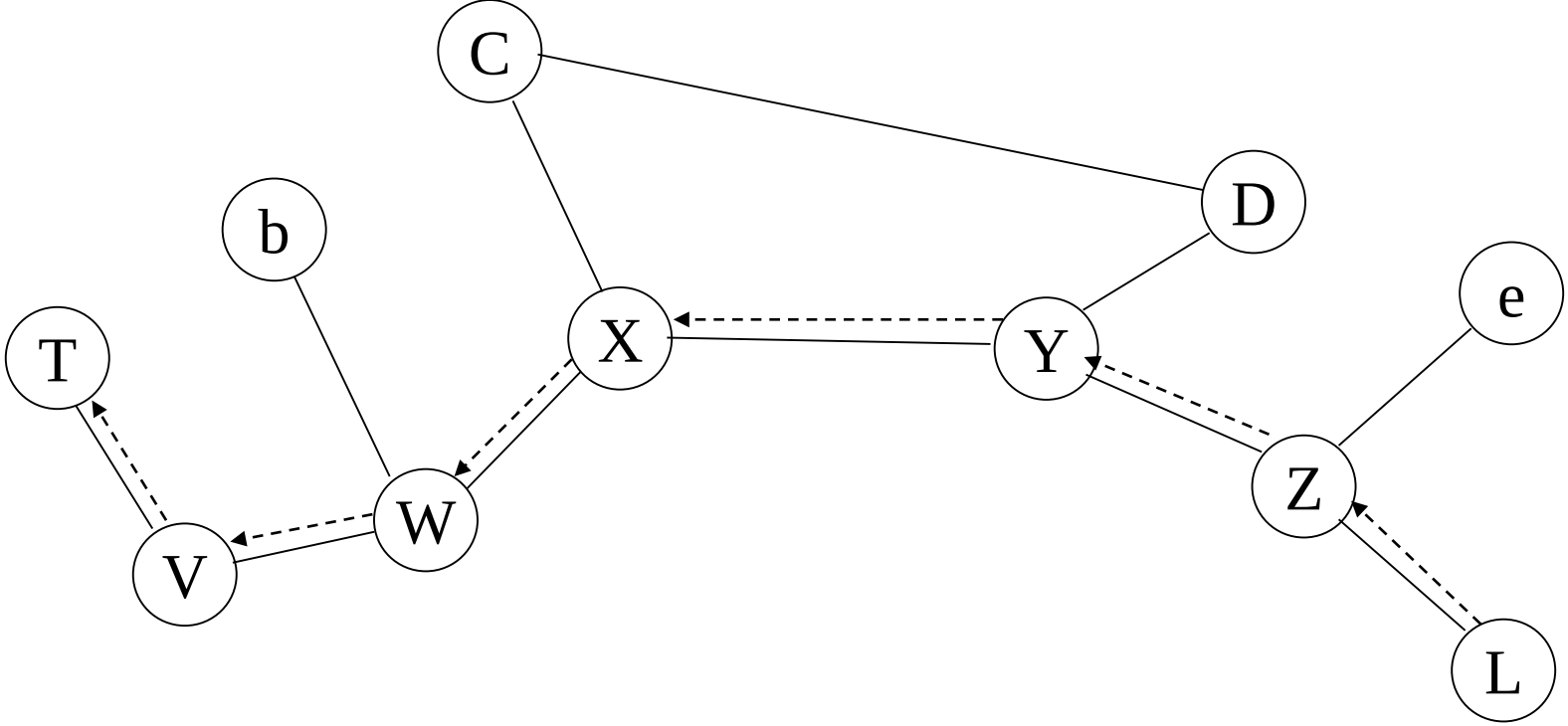
- We can try and limit access of the attacker to the system:
  - Possibly we could constrain the attacker…
    - Try and prevent the attacker from reaching their goal, if we can identify it.
    - Maybe use a “honeypot” to lead the attacker to another part of the system.

Eradication of the attack

- Stop the attack:
  - Probably the easiest thing to do is to terminate the network connection or stop the process.
- We could use a firewall to block the attacker.
- The Intrusion Detection and Isolation Protocol (IDIP) can also block connections.

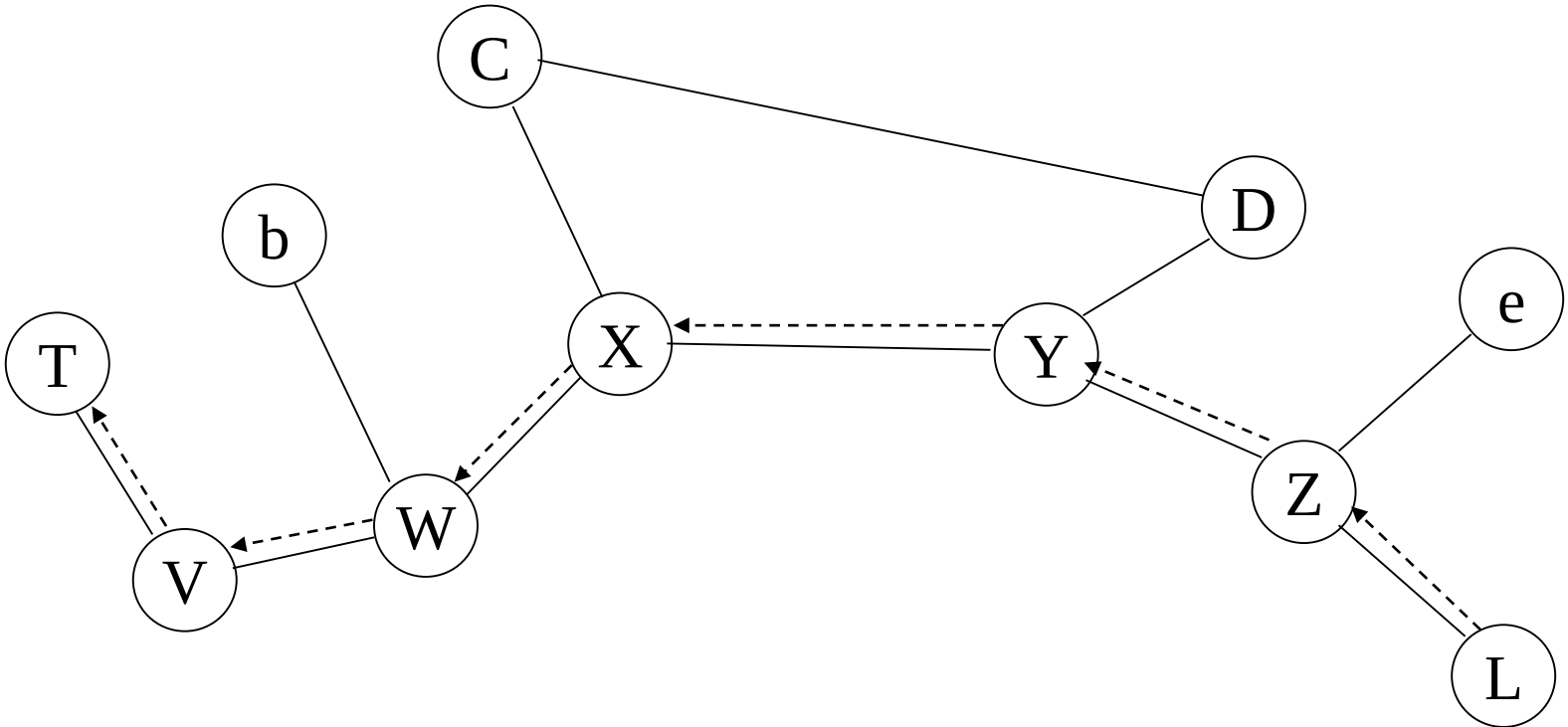
Intrusion Detection and Isolation Protocol (IDIP)

- Kahn and Zurko proposed the use of IDIP as a mechanism for blocking network flooding attacks which would result in denial of service.
- Consider a flooding attack launched against host T, by host L, along the path L→Z→Y→X→W→V→T.
- The flood effectively stops all traffic along that path.
  - V detects the flood, blocks traffic for T, and notifies W.
  - W detects traffic targeting T, stops it, and notifies X.



- X detects the traffic targeting T, stops it, and notifies Y & C.
- W notices the traffic for T has stopped, and it eliminates its suppression.
- T, V, W, and b can know communicate freely again, because X has blocked the flooding.
- Y detects the flooding, stops it and informs Z & D.

– X detects that the traffic going through it for T has stopped, and X stops its suppression.
– Z detects the flooding and suppresses the flooding traffic.
– This process continues until all traffic from L to T is suppressed.



The last two steps…

- **Recovery from the attack**:
  - We restore the system to a secure state.
- **Follow-up to the attack**:
  - Take some action external to the system against the attacker.
  - Pursue legal action against the attacker.
  - Two methods to identify attacker:
    - Thumb printing.
    - IP Header Marking.

IDS Products

- SNORT: Widely used…
- Anzen Flight Jacket
- Axent/Symantec NetProwler and Intruder Alert
- Check Point IPS-1
- Cisco Secure IDS
- CyberSafe Centrax IDS
- Endian Firewall
- Enterasys Dragon IDS
- ISS RealSecure
- Network ICE BlackICE Sentry
- Untangle

SNORT: Signature based.

- Open source – Freeware, but based at SourceFire where there are quite a few professional developers.
- Provides intrusion detection on Windows, Linux, and Unix.
  - http://www.snort.org/
- The current Windows version: ~2.9.7.5
- SNORT allows sets of rules used for detection and logs:
  - Packet information.
  - Alerts.

```
Snort analyzed 523 out of 523 packets, dropping
0(0.000%) packets

Breakdown by protocol:                    Action Stats:
    TCP: 502        (95.985%)             ALERTS: 12
    UDP: 8          (1.530%)              LOGGED: 12
   ICMP: 12         (2.294%)              PASSED: 0
    ARP: 1          (0.191%)
  EAPOL: 0          (0.000%)
   IPv6: 0          (0.000%)
    IPX: 0          (0.000%)
  OTHER: 0          (0.000%)
DISCARD: 0          (0.000%)
```

This is from an old version.

ICMP : Internet Control Message Protocol.

ARP: Address resolution protocol.

EAPOL: EAP over LAN.

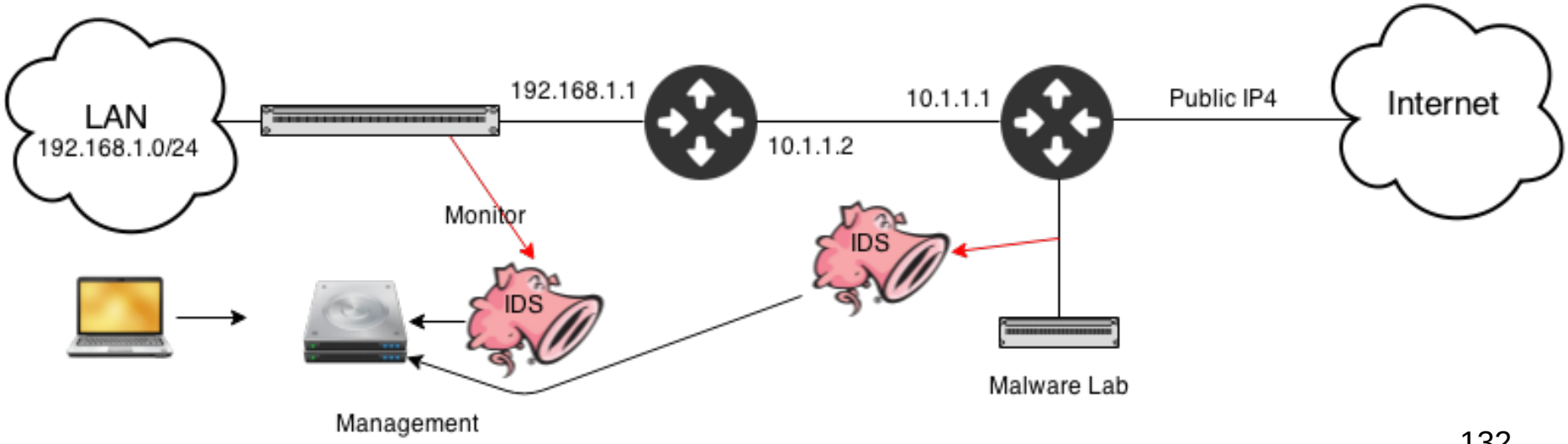
```
Wireless Stats:
Breakdown by type:
    Management Packets: 0          (0.000%)
    Control Packets:    0          (0.000%)
    Data Packets:       0          (0.000%)
===========================================
Fragmentation Stats:
Fragmented IP Packets: 0          (0.000%)
    Fragment Trackers: 0
   Rebuilt IP Packets: 0
   Frag elements used: 0
Discarded(incomplete): 0
   Discarded(timeout): 0
  Frag2 memory faults: 0
===========================================
TCP Stream Reassembly Stats:
        TCP Packets Used: 502        (95.985%)
         Stream Trackers: 6
          Stream flushes: 2
           Segments used: 14
   Stream4 Memory Faults: 0
```

A SNORT Alert

```
[**] [1:620:9] SCAN Proxy Port 8080 attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
08/25-20:28:01.281635 0:50:FC:F1:9E:39 -> 0:60:64:3:DE:7C
type:0x800 len:0x3E
192.168.0.200:1631 -> 130.130.37.205:8080 TCP TTL:128
TOS:0x0 ID:1553 IpLen:20 DgmLen:48 DF
******S* Seq: 0x497BC4F  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Hard to read? → Snorby

- Snorby is a front end for an IDS like Snort.
- https://github.com/Snorby/Snorby
- To set something up at home …
  https://techanarchy.net/2015/01/home-ids-with-snort-and-snorby/
  - From the website, … their infrastructure …



Honeypots

- Honeypots can divert attackers from a critical system, and/or collect information about the attacker's activity.
- They can encourage the attacker to stay on the system long enough for administrators to respond.
  - We can fill a honeypot with fabricated information designed to appear valuable.
  - A legitimate user of the system will not access that location.
  - We fill the system with sensitive monitors and wait! ☺

Possibly **trap-and-trace**.

Honeypots

- The honeypot is a resource that has no production value. There is no legitimate reason for anyone outside the network to interact with a honeypot.
- Any attempt to communicate with the honeypot is most likely a probe, scan, or attack.
- Conversely, if a honeypot initiates outbound communication, the system has probably been compromised.

Intrusion prevention systems

- Obviously rather than just detecting intrusions and working out what happened afterwards, we often want to prevent traffic from getting into our system.
- So, we use intrusion prevention systems,
  - Like firewalls, which will be looked at soon!
  - If you look at what we have just described, it should be clear that we are really talking about access control (again!).

Unified Threat Management Systems

- Since about 2004, the deployment of the mechanisms that we have described somewhat independently, has started to become unified.
  - In unified threat management systems (UTM).
- So we might have **firewalls**, **intrusion detection**, **intrusion prevention**, **gateway anti-virus**, VPN, anti-spam, ..., features together.
- In addition to the consistency advantages, this simplifies the management and maintenance, since it's effectively a single device/software element from a single provider.

- There are a couple of significant disadvantages:
  - It may be a bottleneck in terms of processing and bandwidth.
  - There is a single point of failure.

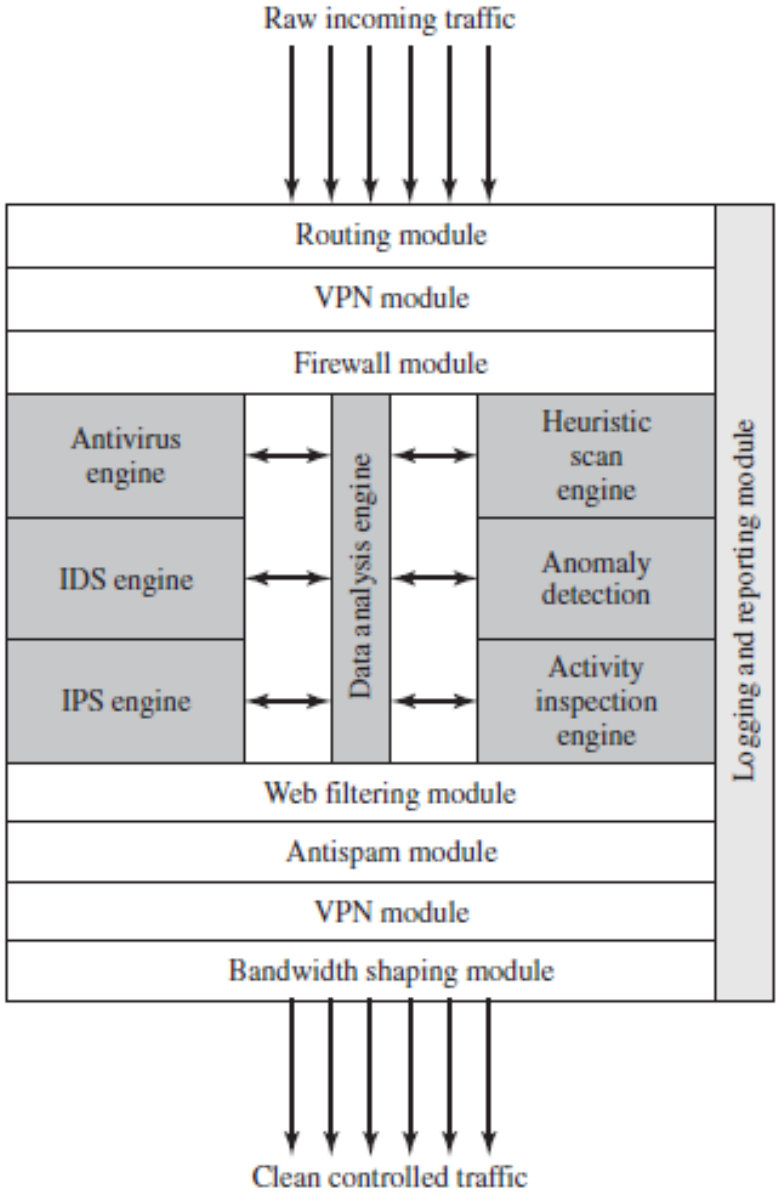


Figure 9.5 **Unified Threat Management Appliance**
Source: Based on [JAME06].

Figure 9.5 Stallings and Brown 2nd edition.

Multiple variables: Recalled

- A collection of measures, (m_1, m_2, …, m_n) and associated weights (w_1, w_2, … w_n).
- Each user has an appropriately determined base profile (M_1, M_2, … M_n).
- To process a system we measure the active profile (μ_1, μ_2, … μ_n).
- We then apply a collection of distance functions D_i to determine $d_i=D_i(M_i,\mu_i)$.

- Our decision can then be based on a threshold, either at the level of individual distances, or in terms of a composite threshold

$$\sum_{i=1}^{n} w_i d_i \leq d_t$$

- It could also be possible to cross-correlate the distance functions.
- The distance functions themselves are functions of the probability distributions associated with the user profile and their actual profile, that is the result of observations.

- Here goes an example …

Total time - average / stdev = answer
answer x weight = login

Event	Average	Stdev	Weight
Logins	4.50	1.25	2
Total time online	287.15	42.12	1
Emails sent	65.40	30.71	1
Orders processed	150.73	20.13	1
Pizza's ordered online	2.03	1.06	0.5

0.4
0.17
0.31
1.95
1,86

Event	Day 1	Day 2
Logins	7	5
Total time online	300	280
Emails sent	60	75
Orders processed	170	190
Pizza's ordered online	2	4

Calculate $\frac{(7-4.5)}{1.25}=2$

So logins contribute
2 * 2 = 4.

slow login = how far from average

CSCI262
System Security (**S7b**)

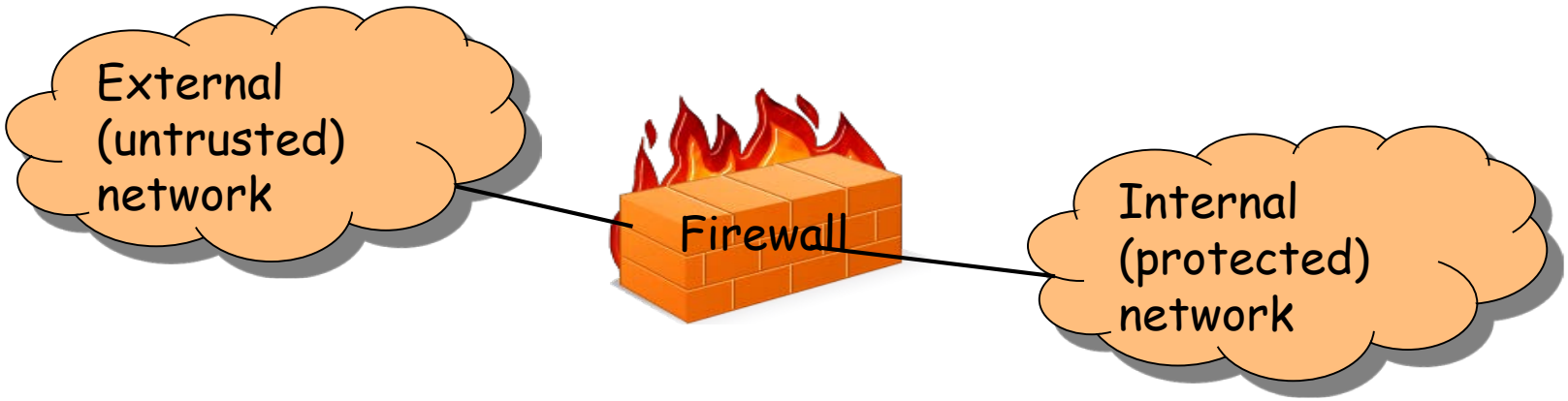
Firewalls

Outline

- What is a firewall?
- Why do I want/need a firewall?
- Firewall Design Principles:
  - Firewall Characteristics.
  - Types of Firewalls.
  - Firewall architectures.

What is a firewall?

- A mechanism or device for controlling connections/traffic between networks.
  - The control can be at different levels, by IP address or content, for example.
- They can effectively enforce some access control and implement security policies.
  - Generally, they are designed to distinguish between the “trusted” internal network and the “distrusted” external network.



Infrastructure motivates firewall use

- There are certain common types of infrastructures:
  - A Centralized data processing system, with a central mainframe supporting a number of directly connected terminals.
  - Local area networks (LANs) interconnecting PCs and terminals to each other.
  - Enterprise-wide network, consisting of multiple, geographically distributed LANs interconnected by a private wide area network (WAN).
  - Etc.

- Consider equipping each workstation and server on the such networks with strong security features, including intrusion protection?
- This could be very expensive, and not necessarily efficient.
- So, the firewall is inserted between the internal network and the Internet to establish a controlled link and to erect an outer security wall or perimeter.
- It is an effective means of protection a local system or network of systems from network-based security threats while affording access to the outside world via WANs or the Internet.

Design requirements

- Access to the network, either in or out, should be through the firewall.
  - Physically the firewall should be the *only* access point in/out of the network.
- Only authorised traffic should be allowed through the firewall.
  - Authorisation is defined with respect to the security policies implemented on the firewall.
- The firewall needs to be “invulnerable”.
- The firewall needs to be trusted.

General control techniques

- Basically: “What can filtering be based on?”:
  - **Service control:** Determines the type of services accessible, based, for example, on IP address and TCP port numbers.
  - **Direction control:** Determines which direction particular services may be requested in.
  - **User control:** Tailors usage to particular users. This is more likely to be applied to internal users, for it to apply to external users appropriate authentication mechanisms would be needed.
  - **Behaviour control:** Within an allowed service particular patterns or structures can be disallowed. This can prevent spam for example.

What can a firewall do?

- A firewall can:
  - Provide a choke point to protect a network from outside. This simplifies security management and deployment.
  - Monitor traffic, in particular for attempted security breaches.
  - Provide a platform for some non-security functions, such as Network Address Translators.
    - That's network layer functionality.

What can’t a firewall do?

- A firewall cannot:
  - Protect against internal attackers.
  - Practically protect against malware or programs infected with malware.
  - Protect against services that bypass the firewall. For example, a dial-in service to a local area network may not pass through the firewall.

Types of firewalls

- Common types of Firewalls:
  - Packet-filtering firewalls.
  - Stateful inspection firewalls or stateful packet filtering.
  - Application-level gateway.
    - Also called proxy servers.
  - MAC layer firewalls.

Packet-filtering firewall

- A packet filtering firewall has a collection of rules.
  - Each incoming and outgoing IP packet is weighed up with respect to the rules, and then either forwarded or discarded.
- The rules are typically based on IP or TCP header fields.
  - If a rule is matched, we determine whether to forward or discard the packet.
  - If there is no match to any rule, then a default action is taken.

The default security policy

- Default discard/reject/deny:
  - If there isn't an explicit rule allowing something, it is blocked.
  - This is a conservative approach. ☺
- Default forward/accept/allow:
  - If there isn't an explicit rule blocking something, it is allowed.
  - This is a liberal approach.☹

Problems with default allow

- How do you know where problems are going to occur?
- How do you protect new applications?
  - Can people just install anything they like and have it run through the firewall?
- On what basis do you restrict access?
  - The security administrator would need to be very up-to-date to keep this secure.

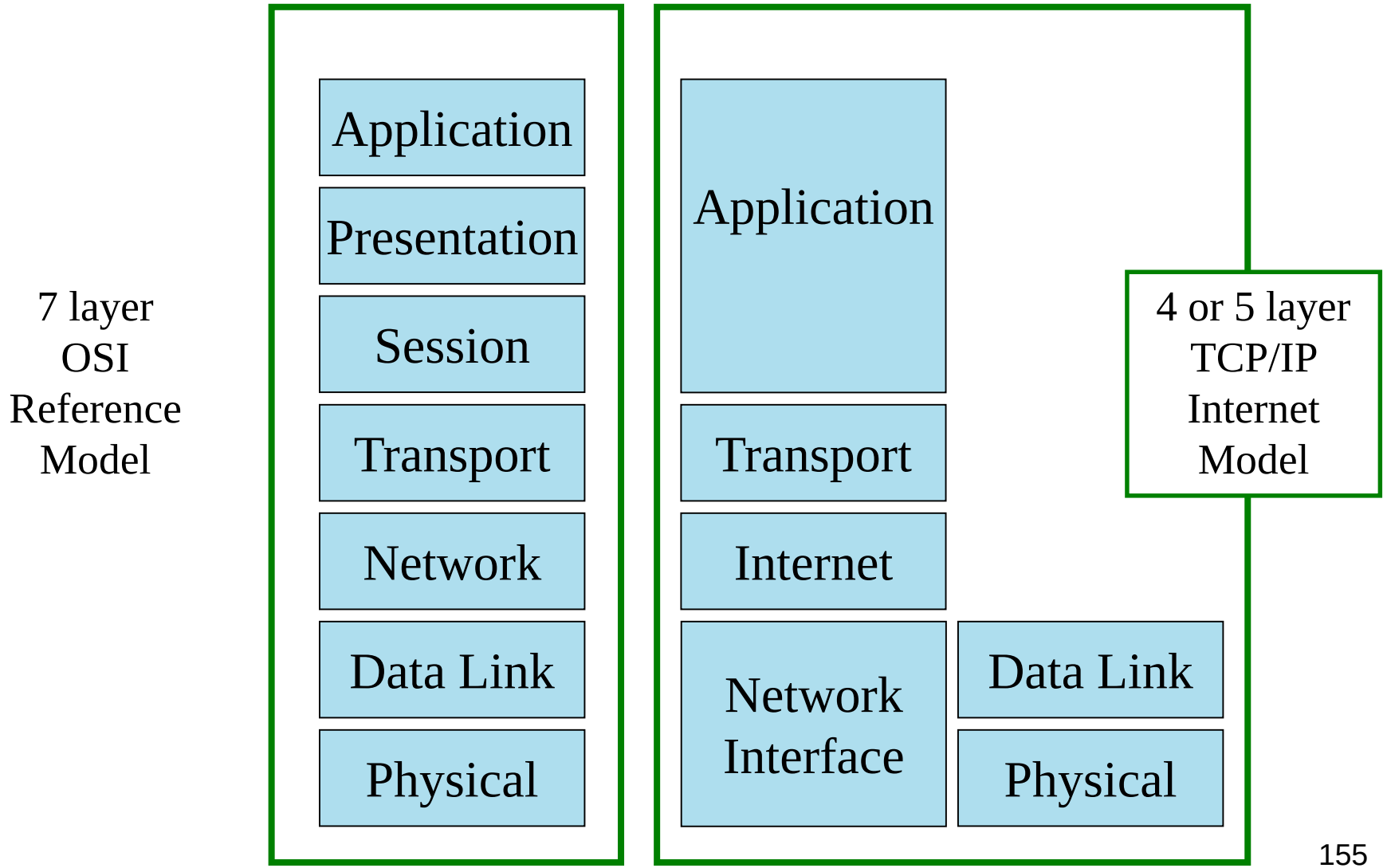
Packet-filtering firewall: Pros and cons

- Packet filtering firewalls are simple and fast.
- They are transparent to users.

But…

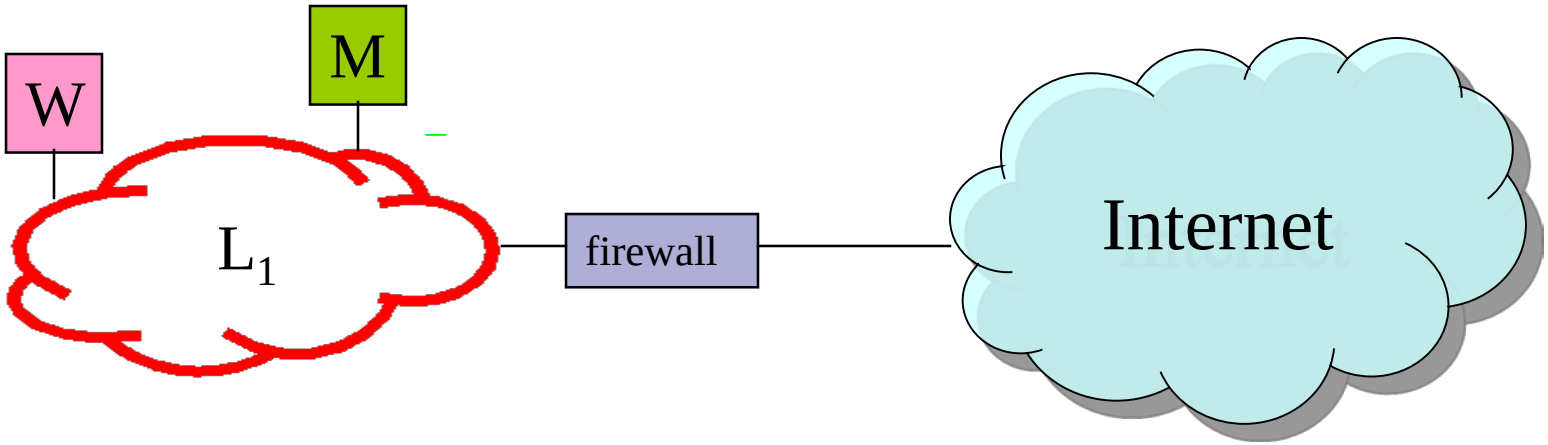
- They don't examine upper-layer data, so cannot prevent, for example, application layer attacks.
  - Logging is limited since there is limited access to the data, which will primarily be upper layer.
  - The lack of upper layer access also typically means they don't support very substantial user authentication, even though it's possible for there to be some
- They are generally vulnerable to TCP/IP based attacks, such as network layer address spoofing.
- Improper configuration can cause security breaches. They are not easy to configure.

OSI versus TCP/IP Model



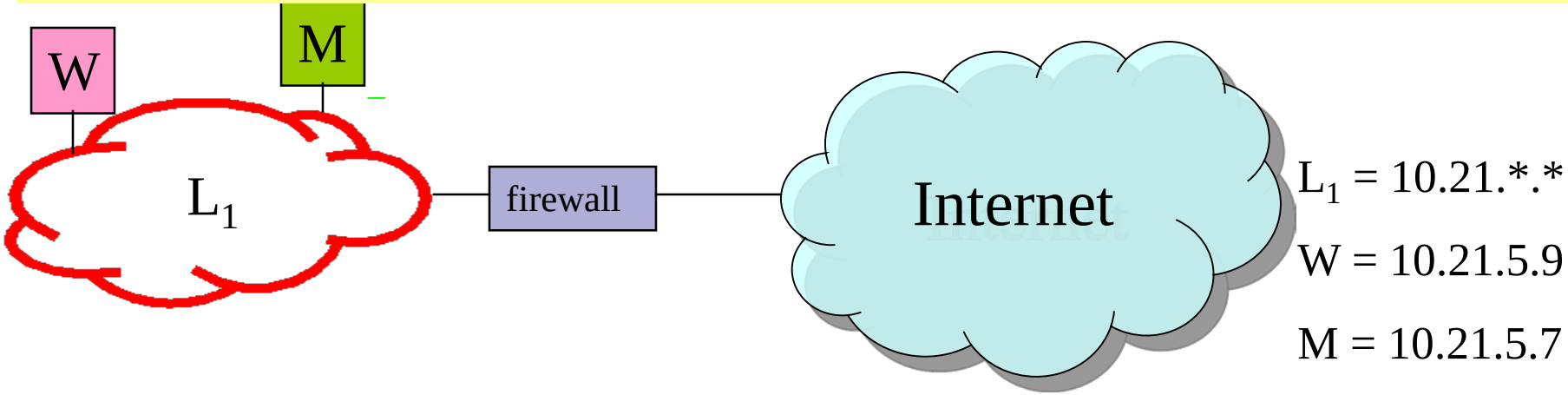
Filters Example

- For network L_1, we would like to allow outsiders:
  1. to access the web server running on W;
  2. to access the SMTP server on M (for e-mail traffic);
  3. to access the DNS server on M.
- We would also like users (programs) on L_1 :
  4. to send e-mail through the SMTP server on M, i.e. allow SMTP server on M to access external SMTP servers;
  5. to access an external NTP server;
  6. We also assume that the DNS server on L_1 can make DNS queries to external DNS servers.
- All other traffic between L_1 and the external networks are not allowed.



Filters Example (cont'd)

	INBOUND traffic	OUTBOUND traffic
1.	request to Web server on W	response from Web server on W
2.	request to SMTP server on M	response from SMTP server on M
3.	request to DNS server on M	response from DNS server on M
4.	response from external SMTP server from SMTP server on M	request from SMTP server on M to external SMTP server
5.	response from external NTP server	request to external NTP servers
6.	response from external DNS server to DNS server on M	request from DNS server on M to external DNS server



Filters Example (cont'd)

- To achieve 1 the following filtering rules can be used:

Rule	Direction	Source Address	Dest. Address	Protocol	Source Port	Dest. Port	ACK Set	Action
A	In	Any	10.21.5.9	TCP	>1023	80	Either	Permit
B	Out	10.21.5.9	Any	TCP	80	>1023	Yes	Permit
C	Either	Any	Any	Any	Any	Any	Either	Deny

Note: In practice, clients should have unprivileged source port of *1024* or higher. The allowable source ports can be written as >1023.

The ack bit is reset (=0) only in TCP connection set-up requests. In all other TCP segments, ack is set.
Rule C is default deny

Static vs dynamic packet filtering

- In static packet filtering rules are developed prior to installation and installed with the firewall, and subsequently changed by direct human input.
- In dynamic packet filtering, the firewall is able to respond to events and change the rules as appropriate.

Stateful Inspection firewalls

- Sometimes it can be useful if the filtering firewall can keep state information.
- Useful for filtering traffic built on stateless protocol such as UDP.
- Example:
  - Allow UPD packet in (carrying a response) only as a response to a previous outgoing UPD packet (carrying a request).
  - Firewall has to remember outgoing UPD packet.
- Can be expensive and complicates administration
  - e.g., State information need be consistently maintained

Application-level gateway (or proxy servers)

- These act as relays for application level traffic, often of limited types, such as web traffic.
  - These act on the application layer of the TCP/IP stack, application, session or presentation in OSI.
- End-to-end connections between server and client are not formed.
- Outgoing and incoming traffic are both inspected.
  - Checks application content for appropriateness, such as restricting particular websites.
  - Checks the format for protocol data.
    - In particular it can protect against malformed IP or TCP packets.

MAC layer firewalls

- Operate at the data link layer.
  - Actually at media access control sub-layer.
- Typically specify the specific traffic allowed from/to a network interface card, or something similar.

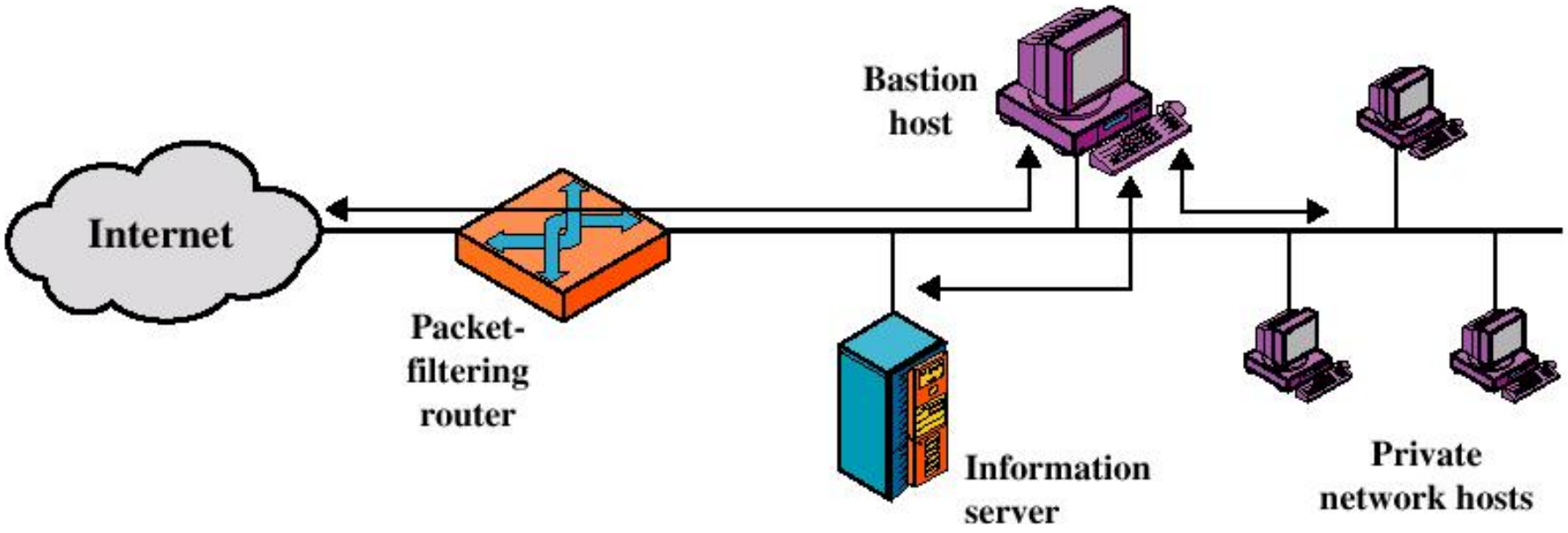
Firewall Architecture

- Packet-filtering routers:
  - Add firewall functionality to a border router.
- Single homed bastion host.
- Double or dual-homed bastion hosts.
- Screened subnet firewall.

Bastion Hosts

- These are hosts identified by the firewall administrator as critical points in the security of a network.
- They typically have limited functionality, to reduce exposure to vulnerabilities and improve performance, and serve as a platform for an application-level gateway.
- The way in which the bastion host performs, and is fitted with other parts of the system, determines the firewall architecture.

Single-homed bastion host

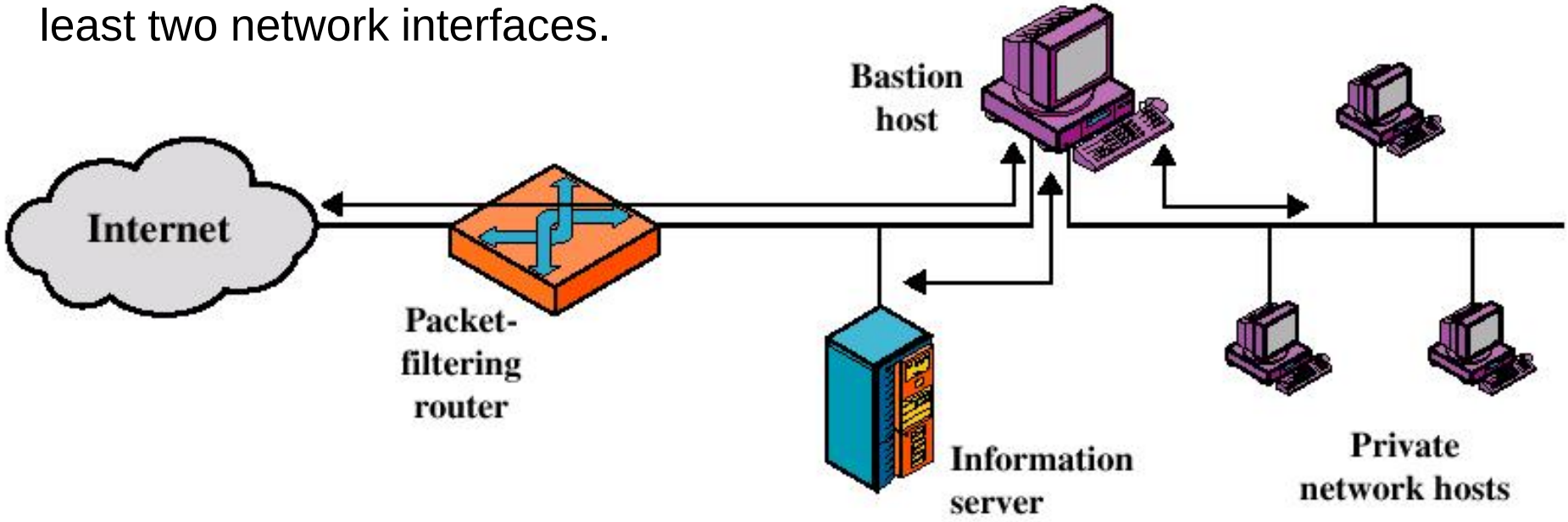


- The firewall is a composite of two systems:
  - A packet-filtering router which pre-filters to reduce the load on the second unit.
  - A bastion host (probably an application-layer firewall).
    - Provides proxy access for the inside.

- The bastion host can perform authentication and proxy functions that the packet-filtering router (firewall) cannot, because the proxy can see the application level information.
- This improves on single configurations:
  - We have both packet-level and application-level filtering.
    - This gives flexibility in defining security policies.
  - An intruder will needed to get through two separate systems.
- There is also flexibility in providing direct Internet access to, for example, a web server with public information.

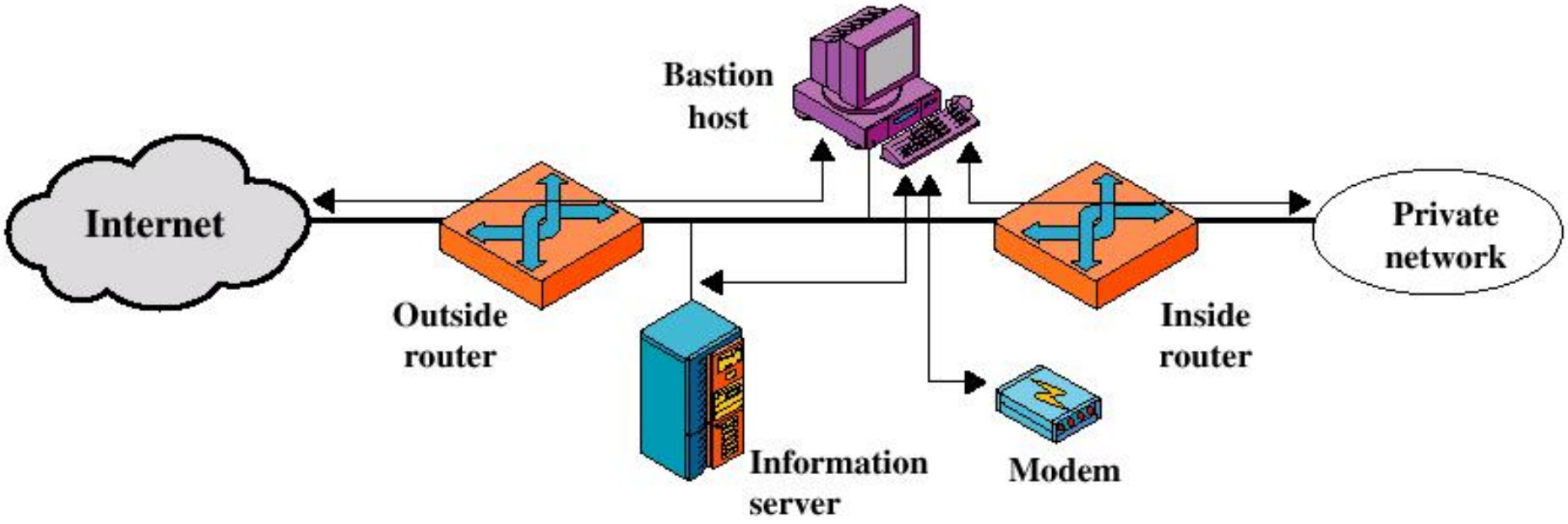
Dual-homed bastion host

This architecture is built around a dual-homed host, i.e. something with at least two network interfaces.



- One interface connects to the external network, another to the internal network.
- Everything (in/out) goes through the bastion host now.

Screened-subnet firewall system



- Screened subnet firewall configuration.
  - This is the most expensive, and the most common setting
  - Two packet-filtering routers are used.
  - Creation of an isolated sub-network, the DMZ.