

# Evading IDS, Firewalls, and Honeypots

## Module 16

Engineered by **Hackers**. Presented by Professionals.



# SECURITY NEWS



## INTERNATIONAL BUSINESS TIMES

November 29, 2010 9:36 PM

### How Wikileaks uses technology to protect anonymity of whistle-blowers

Wikileaks works on a model that allows whistleblowers to submit leaked documents through internet or postal mail, ensuring that the sender's identity is concealed and trails cleared.

The network used by Wikileaks is similar to a technology called The Onion Router or Tor. Tor is open-source software and its website states that it is currently used by a branch of U.S. Navy for gathering intelligence.

It is a system used to outflank filtering and censors enabling users to evade blockers keeping their identity anonymous. To evade online traffic analysis Tor "distributes transactions over several places on the Internet, so no single point can link you to your destination. The idea is similar to using a twisty, hard-to-follow route in order to throw off somebody who is tailing you - and then periodically erasing your footprints. Instead of taking a direct route from source to destination, data packets on the Tor network take a random pathway through several relays that cover your tracks so no observer at any single point can tell where the data came from or where it's going."

<http://uk.ibtimes.com>



Copyright © by EC-Council  
All Rights Reserved. Reproduction is Strictly Prohibited.

# Module Objectives

- Intrusion Detection Systems (IDS)
- Ways to Detect an Intrusion
- Types of Intrusion Detection Systems
- Firewall
- Types of Firewall
- Firewall Identification Techniques
- Honeypot
- Types of Honeypot



- How to Set up a Honeypot?
- IDS, Firewall and Honeypot System
- Evading IDS
- Evading Firewall
- Detecting Honeypots
- Firewall Evading tools
- Countermeasures
- Firewall and IDS Penetration Testing



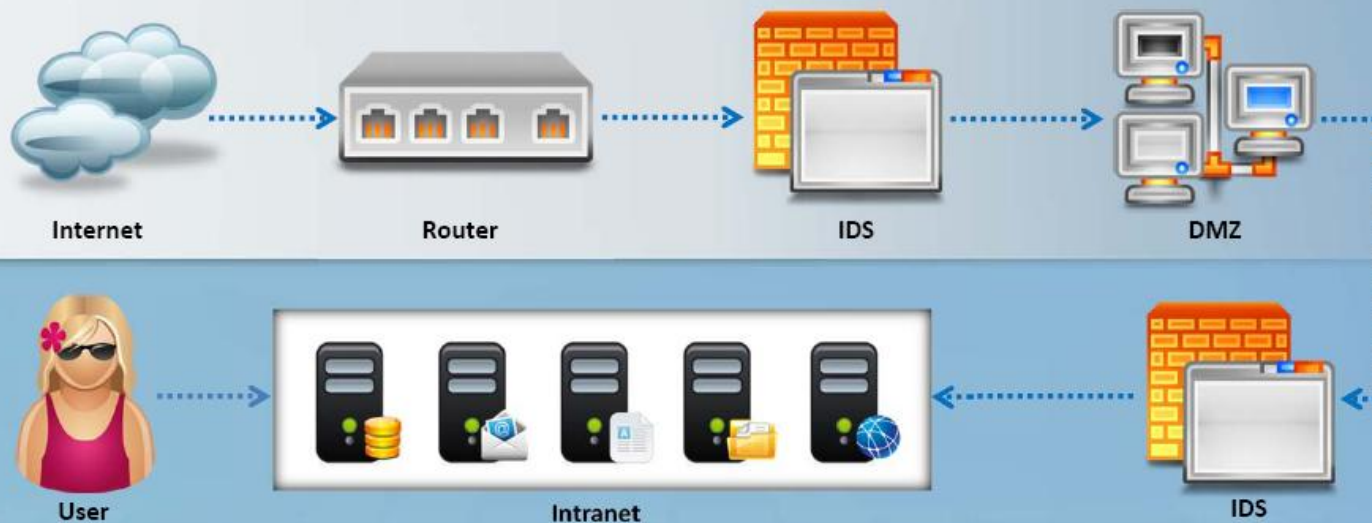
Copyright © by EC-Council  
All Rights Reserved. Reproduction is Strictly Prohibited.

# Module Flow



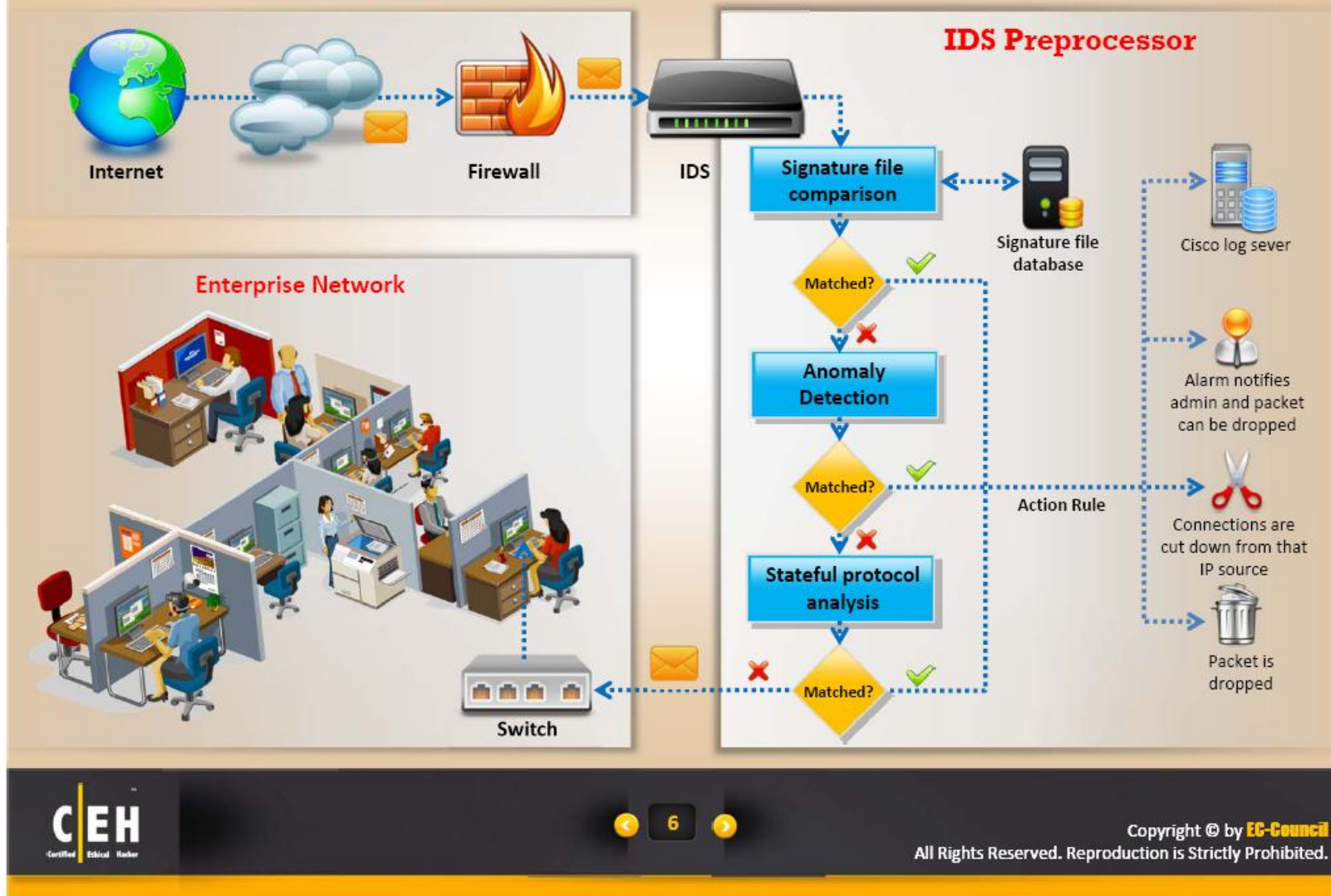


# Intrusion Detection Systems (IDS) and their Placement



- An intrusion detection system (IDS) **gathers and analyzes information** from within a computer or a network, to **identify** the possible violations of security policy, including unauthorized access, as well as misuse
- An IDS is also referred to as a **"packet-sniffer,"** which intercepts packets traveling along various communication mediums and protocols, usually TCP/IP
- The packets are analyzed after they are **captured**
- An IDS evaluates a **suspected intrusion** once it has taken place and signals an alarm

# How IDS Works?



# Ways to **Detect** an Intrusion

Three ways to detect an intrusion:

## Signature Recognition

It is also known as misuse detection. Signature recognition tries to identify events that misuse a system



## Anomaly Detection

It detects the intrusion based on the fixed behavioral characteristics of the users and components in a computer system



## Protocol Anomaly Detection

In this type of detection, models are built on TCP/IP protocols using their specifications





# Types of **Intrusion Detection Systems**

## Network-based Intrusion Detection

- These mechanisms typically consist of a black box that is placed on the network in the promiscuous mode, listening for patterns indicative of *an intrusion*



## Host-based Intrusion Detection

- These mechanisms usually include auditing for events that occur on a specific host
- These are not as common, due to the overhead they incur by having to monitor each system event



## Log File Monitoring

- These mechanisms are typically programs that parse log files after an event has already occurred, such as failed log in attempts



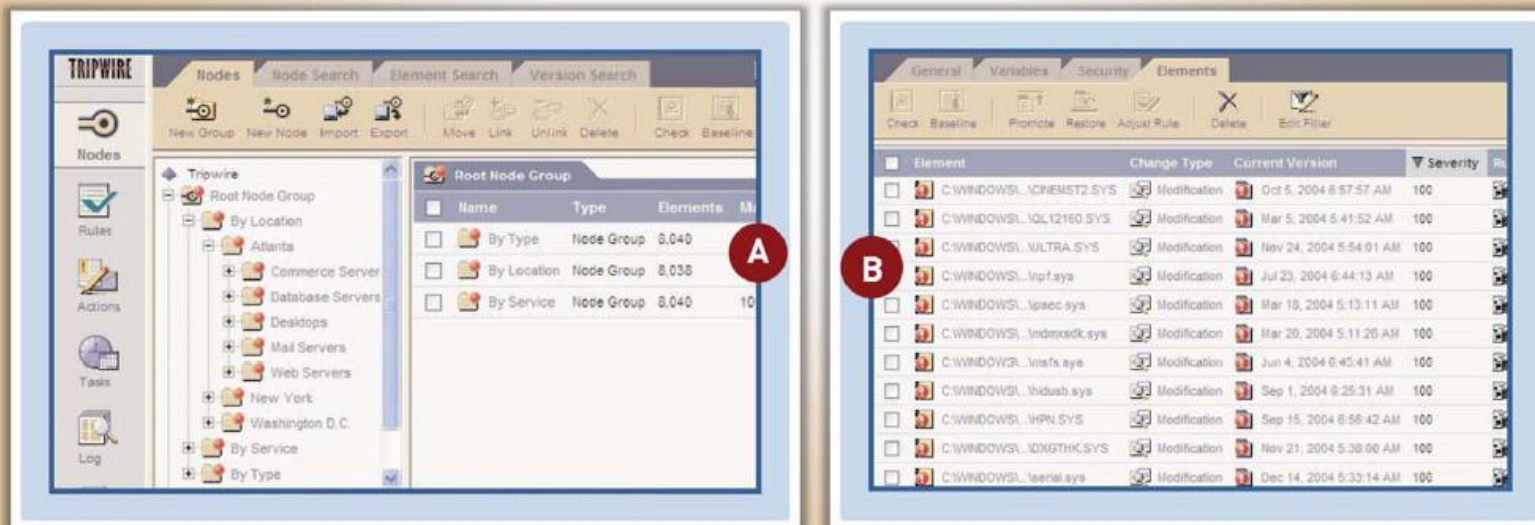
## File Integrity Checking

- These mechanisms check for Trojan horses, or files that have otherwise been modified, indicating an intruder has already been there, for example, Tripwire



# System Integrity Verifiers (SIV)

Tripwire is a **System Integrity Verifiers (SIV)** that monitors system files and detects changes by an intruder



<http://www.tripwire.com>

# General **Indications** of Intrusions

## File System Intrusions

- The presence of new, unfamiliar files, or programs
- Changes in file permissions
- Unexplained changes in the file's size
- Rogue files on the system that do not correspond to your master list of signed files
- Unfamiliar file names in directories
- Missing files



## Network Intrusions

- Repeated probes of the available services on your machines
- Connections from unusual locations
- Repeated log in attempts from remote hosts
- Arbitrary data in log files, indicating an attempt at creating either a Denial of Service, or a crash service





# General **Indications** of System Intrusions



Modifications to system software and configuration files



Gaps in the system accounting



Unusually slow system performance



System crashes or reboots



Short or incomplete logs



Missing logs or logs with incorrect permissions or ownership



Unfamiliar processes



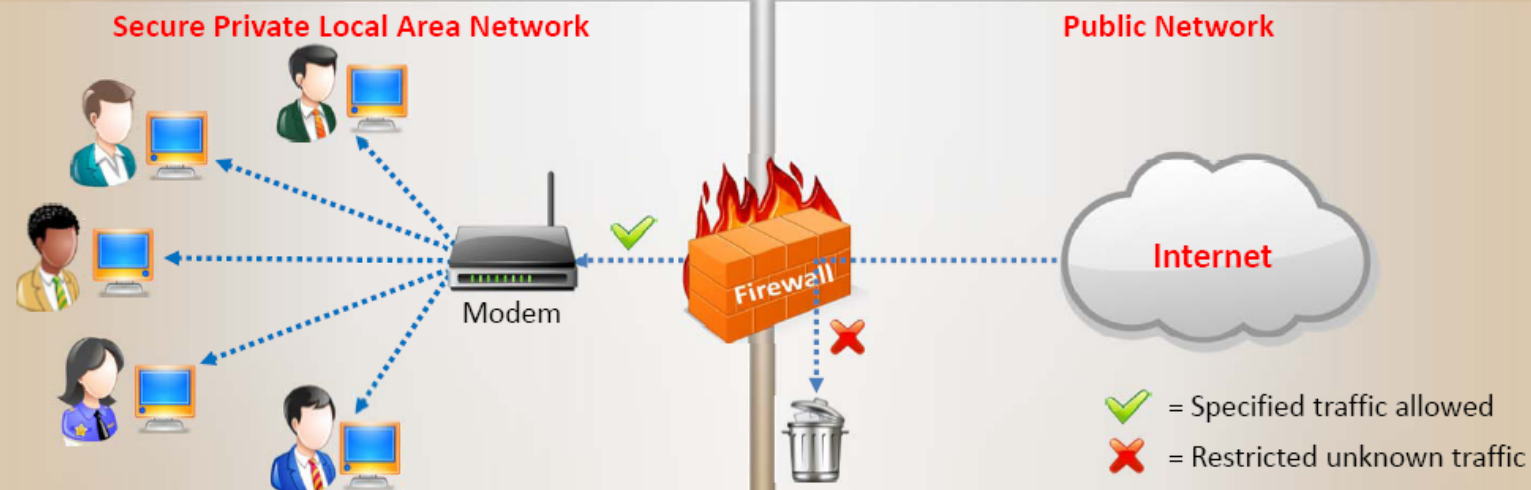
Unusual graphic displays or text messages



# Firewall

- Firewall is a hardware or software or combination of both designed **to prevent unauthorized access** to or from a private network
- It is placed at the **junction point, or gateway** between the two networks, which is usually a private network and a public network such as the **Internet**

- Firewall **examines all messages entering or leaving the intranet** and blocks those that do not meet the specified security criteria
- Firewalls may be concerned with the **type of traffic** or with the **source or destination addresses and ports**



# Firewall Architecture

## Bastion Host:

- Bastion host is a computer system designed and configured to protect **network resources** from attack
- Traffic entering or leaving the network passes through the firewall, it has two interfaces:
  - public interface** directly connected to the Internet
  - private interface** connected to the intranet



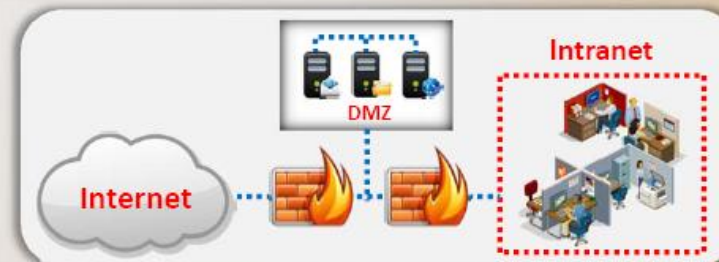
## Screened subnet:

- The screened subnet or DMZ (additional zone) contains **hosts** that offer public services
- Public zone is directly connected to the Internet and has **no hosts controlled** by the organization
- Private zone has systems that **Internet users** have no business accessing



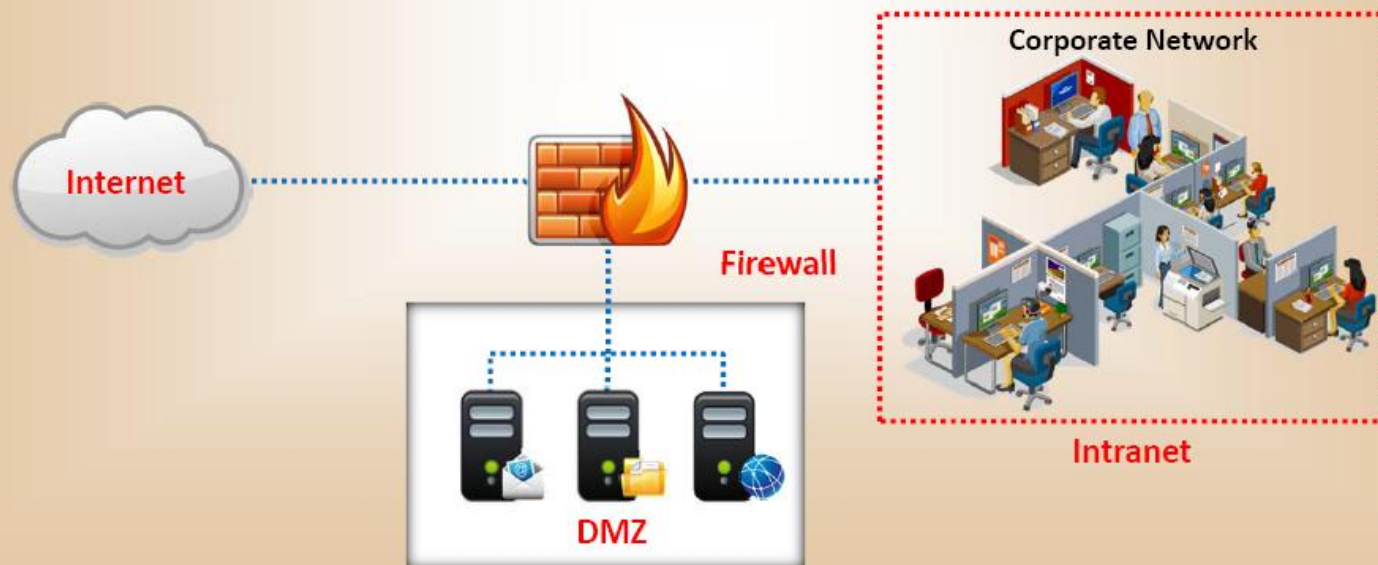
## Multi-homed firewall:

- In this case, **more than three interfaces** are present that allow for further subdividing the systems based upon the **specific security objectives** of the organization



# DeMilitarized Zone (DMZ)

- DMZ is a network that **serves as a buffer** between the internal secure network and insecure internet
- It is created using **firewall with three or more network interfaces** assigned with specific roles such as Internal trusted network, DMZ network, and External un-trusted network (Internet)



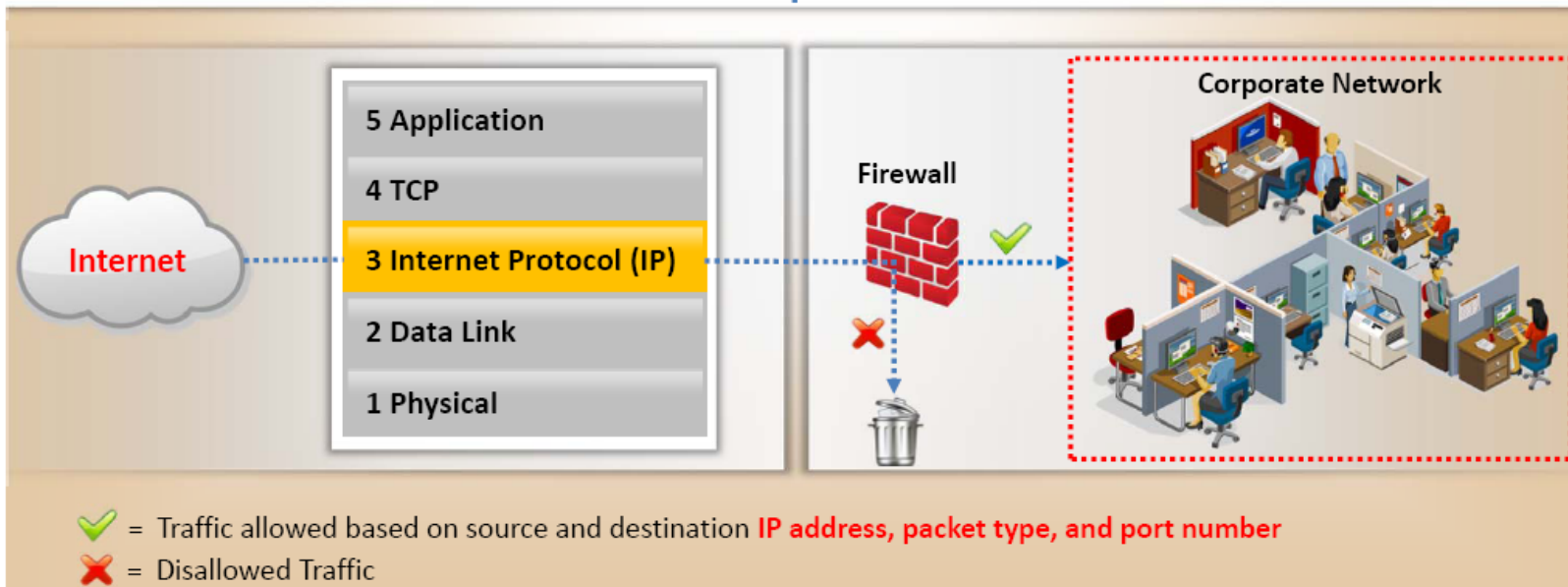


# Types of Firewall



# Packet Filtering Firewall

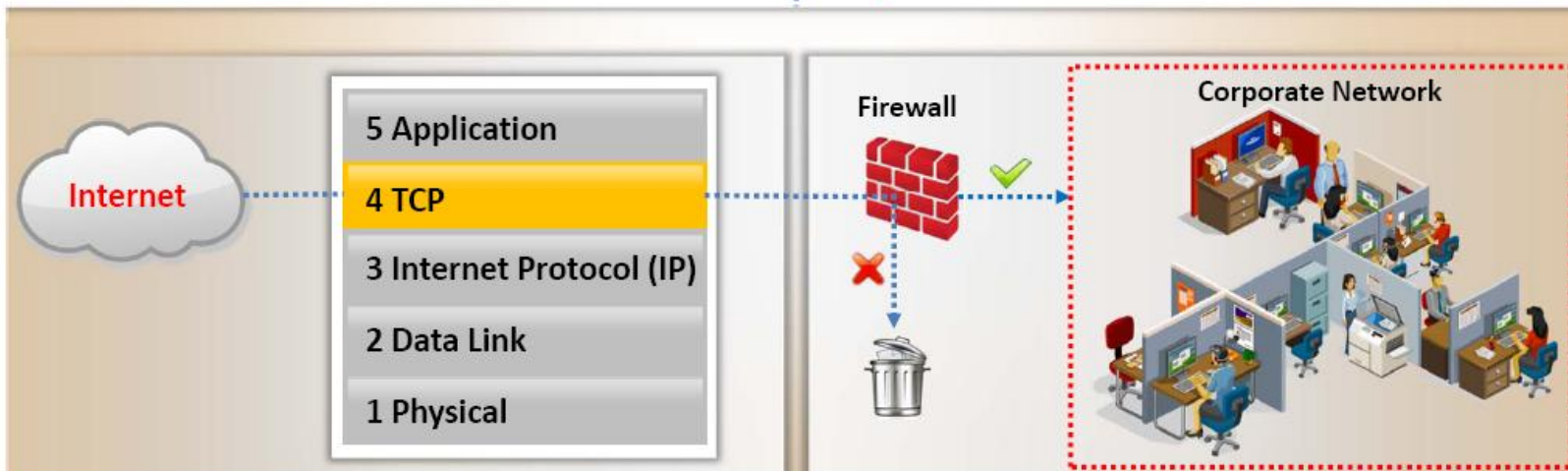
- Packet filtering firewalls work at the **network level of the OSI model** (or the IP layer of TCP/IP), they are usually a part of a router
- In a packet filtering firewall, **each packet is compared** to a set of criteria before it is forwarded
- Depending on the **packet and the criteria**, the firewall can:
  - Drop the packet
  - Forward it, or send a message to the originator
- Rules can include the source and the destination **IP address**, the source and the destination **port number**, and the **protocol** used





# Circuit-Level Gateway Firewall

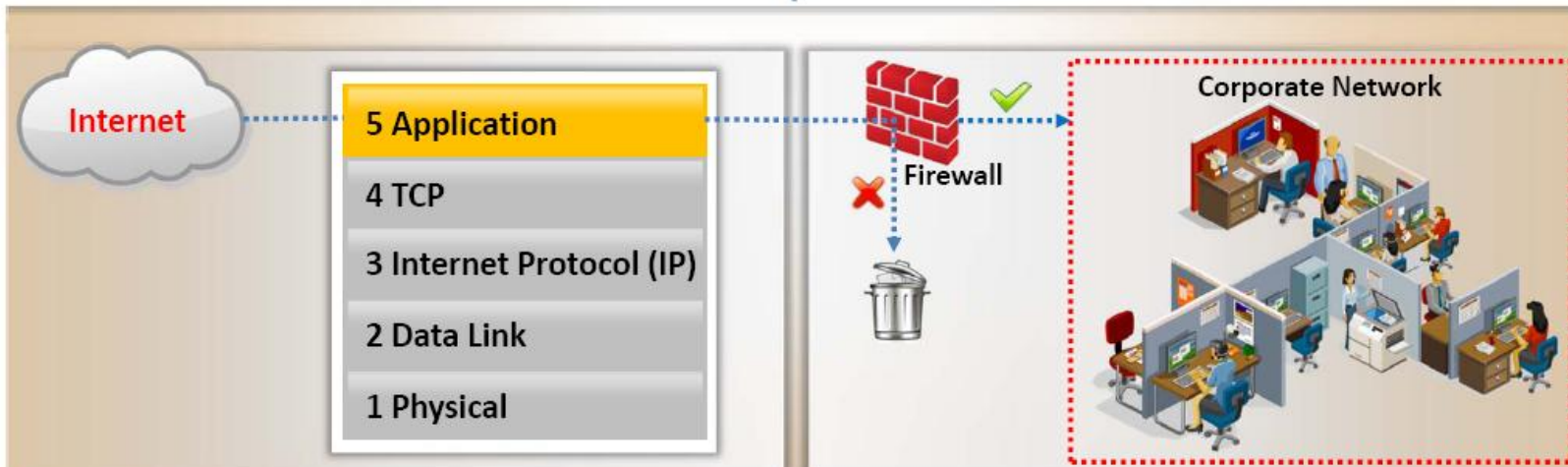
- Circuit-level gateways work at the **session layer of the OSI model** or the TCP layer of TCP/IP
- They monitor **TCP handshaking between packets** to determine whether a requested session is legitimate
- Information passed to a **remote computer** through a circuit-level gateway appears to have originated from the gateway
- Circuit-level gateways **hide information** about the private network they protect, but they **do not filter individual packets**



- ✓ = Traffic allowed based on **session rules**, such as when a session is initiated by a recognized computer
- ✗ = Disallowed Traffic

# Application-Level Firewall

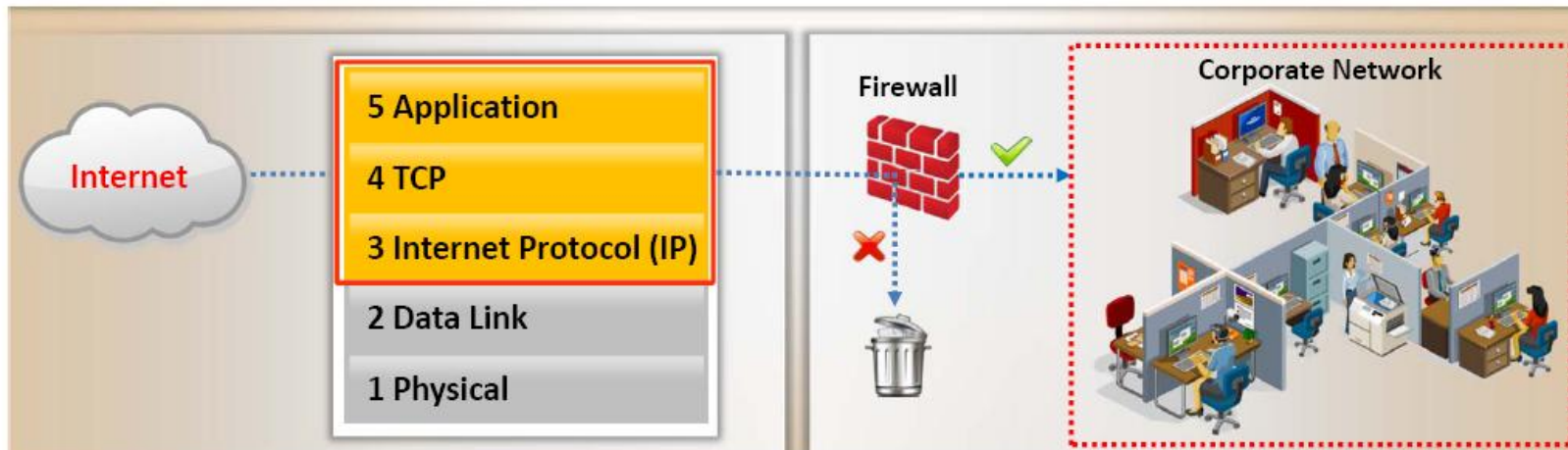
- Application-level gateways (proxies) can filter packets at the **application layer of the OSI model**
- Incoming or outgoing packets **cannot access services** for which there is **no proxy**
- An application-level gateway that is **configured to be a web proxy** will not allow any FTP, gopher, telnet, or other traffic
- As an application-level gateway examines packets at an application layer, it can filter **application specific commands** such as http:post and get



- ✓ = Traffic allowed based on **specified applications** (such as a browser) or a **protocol**, such as FTP, or combinations
- ✗ = Disallowed Traffic

# Stateful Multilayer Inspection Firewall

- Stateful multilayer inspection firewalls **combine the aspects of the other three types** of firewalls
- They **filter packets at the network layer**, to determine whether session packets are legitimate, and they evaluate the contents of packets at the application layer



- ✓ = Traffic is filtered at 3 layers based on a wide range of the **specified application, session, and packet filtering rules**
- ✗ = Disallowed Traffic



# Firewall Identification: **Port Scanning**

Port scan helps the attacker find which ports are available (i.e., what service might be listening to a port); it consists of sending a message to each port, one at a time

The kind of response received indicates **whether the port is used** and can therefore be probed further for weakness

Some firewalls **will uniquely identify themselves** using simple port scans

For example: Check Point's FireWall-1 listens on TCP ports 256, 257, 258, and 259 and Microsoft's Proxy Server usually listens on TCP ports 1080 and 1745



# Firewall Identification:

## Firewalking

It is a technique for testing the vulnerability of a firewall and mapping the routers of a network that are behind a firewall



Firewalking is similar to tracerouting and works by sending TCP or UDP packets into the firewall that have a TTL set at one hop greater than the targeted firewall

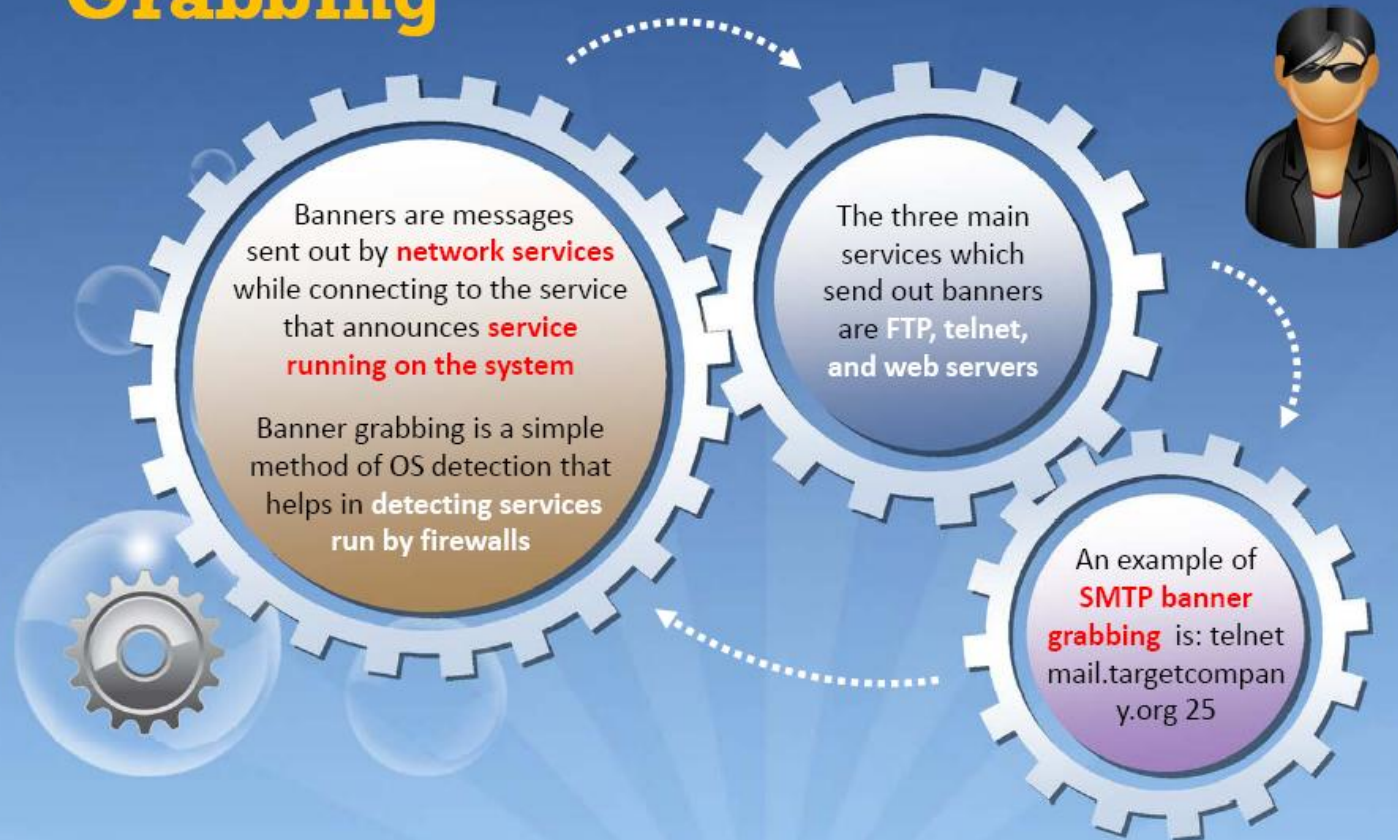
If the packet makes it through the gateway, it is forwarded to the next hop where the TTL equals zero and elicits a TTL "exceeded in transit" message, at which point the packet is discarded



Using this method, access information on the firewall can be determined if successive probe packets are sent



# Firewall Identification: **Banner Grabbing**



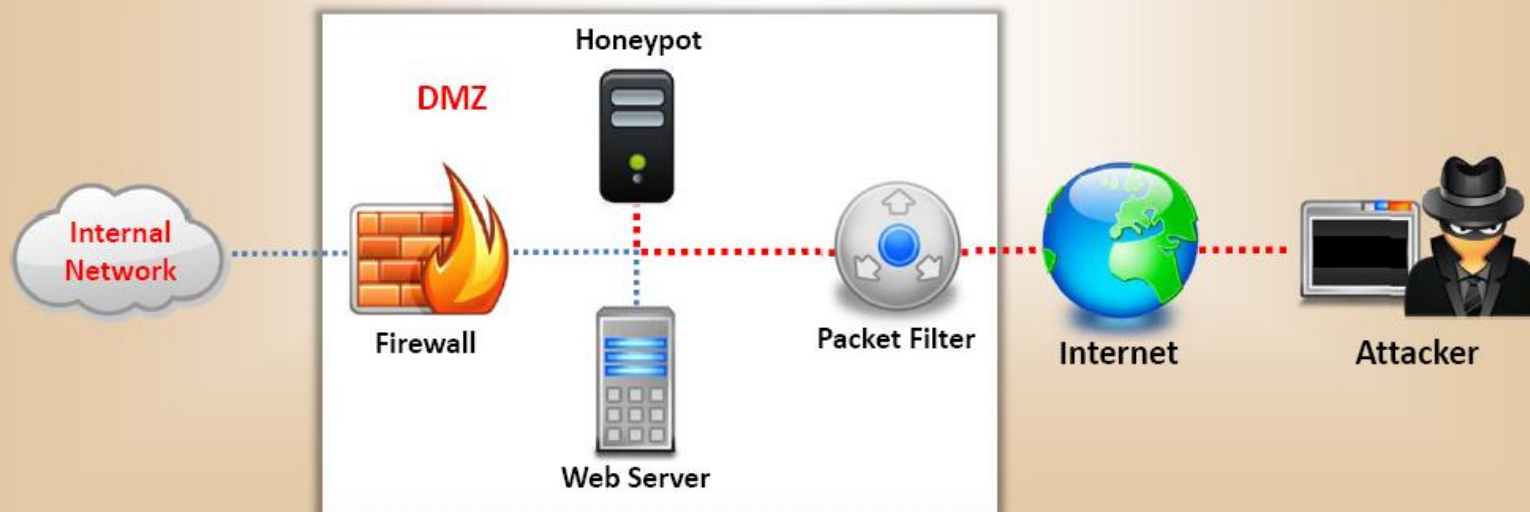


# Honeypot

Honeypot is an information system resource that is expressly **set up to attract and trap people** who attempt to penetrate an organization's network

It has no authorized activity, does not have any production value and is **likely a probe, attack, or compromise**

A honeypot can be used to **log access attempts** to those ports including the attacker's keystrokes. This could send early warnings of a more concerted attack



# Types of Honeypots

## Low-interaction honeypot

- They work by emulating services and programs that would be found on an individual's system
- If the attacker does something that the emulation does not expect, the honeypot will simply generate an error
- Captures limited amounts of information, mainly transactional data and some limited interaction.
- Ex: Specter, Honeyd, and KFSensor



## High-interaction honeypot

- Entire system or network of computers, to have a controlled area in which the attackers can interact with real applications and programs
- Rely on the border devices to control traffic so that attackers can get in, but outbound activity is tightly controlled
- Captures far more information, including new tools, communications, or attacker keystrokes
- Ex: Symantec Decoy Server and Honeynets



# How to Set Up a Honeytrap?



- 1 Download or purchase a honeypot software
- 2 Tiny Honeytrap, LaBrea, and Honeyd are some of the programs available for Linux systems
- 3 KFSensor is one software that works with Windows



- 1 Log in as an administrator on the computer to install a honeypot on to the computer



- 1 Install the software on your computer
- 2 Choose the "Full Version" to make sure every feature of the program is installed



# How to Set Up a Honeypot?



- Place the honeypot software in the **"Program Files"** folder
- Once you have chosen the folder, click **"OK"** and the program will install

4



- Restart** your computer for the honeypot to work

5



- Configure** the honeypot to check the items that you want the honey pot to watch for, including **services, applications and Trojans**, and name your domain

6

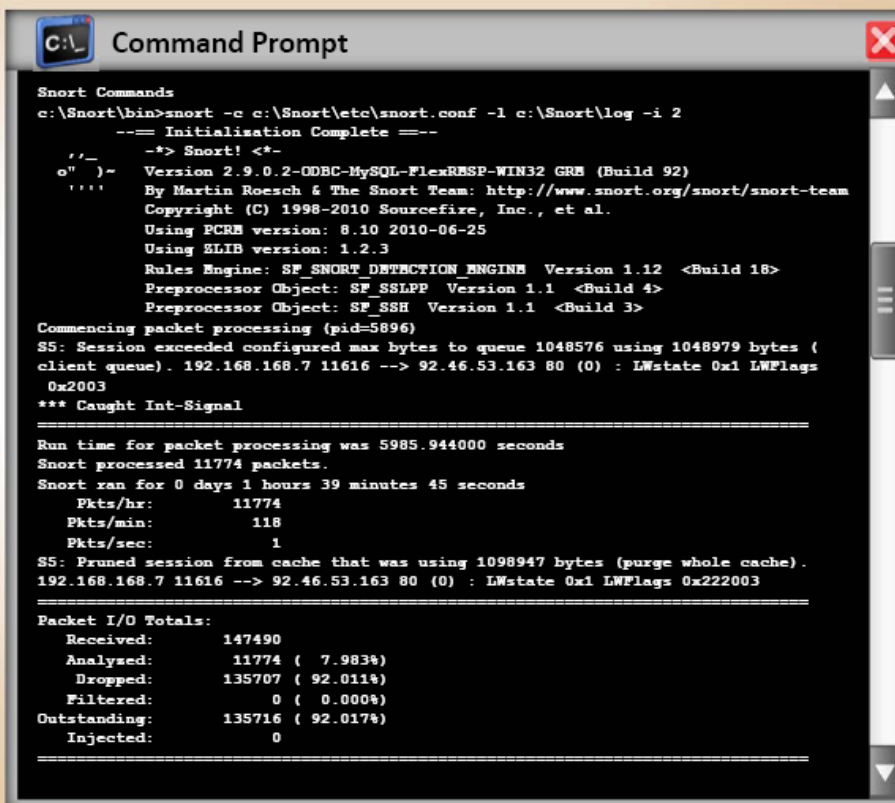


# Module Flow



# Intrusion Detection Tool: Snort

- Snort is an open source network intrusion detection system, capable of performing real-time **traffic analysis and packet logging on IP networks**
- It can perform **protocol analysis** and **content searching/matching**, and is used to detect a variety of **attacks and probes**, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, and OS fingerprinting attempts
- It uses a flexible **rules language** to describe traffic that it should collect or pass, as well as a **detection engine** that utilizes a modular plug-in architecture
- **Uses** of Snort:
  - Straight packet sniffer like tcpdump
  - Packet logger (useful for network traffic debugging, etc.)
  - Network intrusion prevention system



```
C:\> Command Prompt

Snort Commands
c:\Snort\bin>snort -c c:\Snort\etc\snort.conf -l c:\Snort\log -i 2
--== Initialisation Complete ==--
--> Snort! <*-
    _ _
    o" )~
    ' ' '
    Version 2.9.0.2-ODBC-MySQL-FlexRMBP-WIN32 GRM (Build 92)
    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
    Copyright (C) 1998-2010 Sourcefire, Inc., et al.
    Using PCRE version: 8.10 2010-06-25
    Using ELIB version: 1.2.3
    Rules Engine: SF_SNORT DETECTION ENGINE Version 1.12 <Build 18>
    Preprocessor Object: SF_SSLPP Version 1.1 <Build 4>
    Preprocessor Object: SF_SSH Version 1.1 <Build 3>

Commencing packet processing (pid=5896)
SS: Session exceeded configured max bytes to queue 1048576 using 1048979 bytes (
client queue). 192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags
0x2003
*** Caught Int-Signal

Run time for packet processing was 5985.944000 seconds
Snort processed 11774 packets.
Snort ran for 0 days 1 hours 39 minutes 45 seconds
Pkts/hr: 11774
Pkts/min: 118
Pkts/sec: 1
SS: Pruned session from cache that was using 1098947 bytes (purge whole cache).
192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags 0x222003

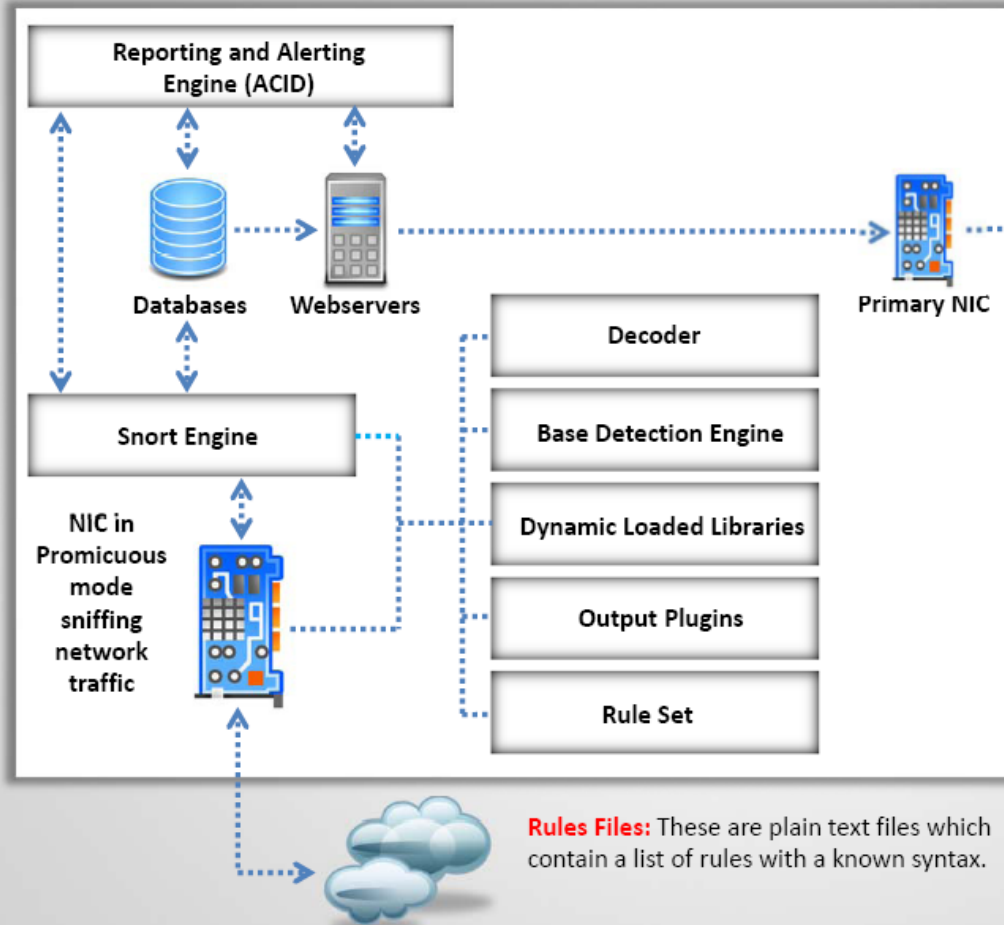
Packet I/O Totals:
Received: 147490
Analysed: 11774 ( 7.983%)
Dropped: 135707 ( 92.011%)
Filtered: 0 ( 0.000%)
Outstanding: 135716 ( 92.017%)
Injected: 0
```

<http://www.snort.org>





# How Snort Works?



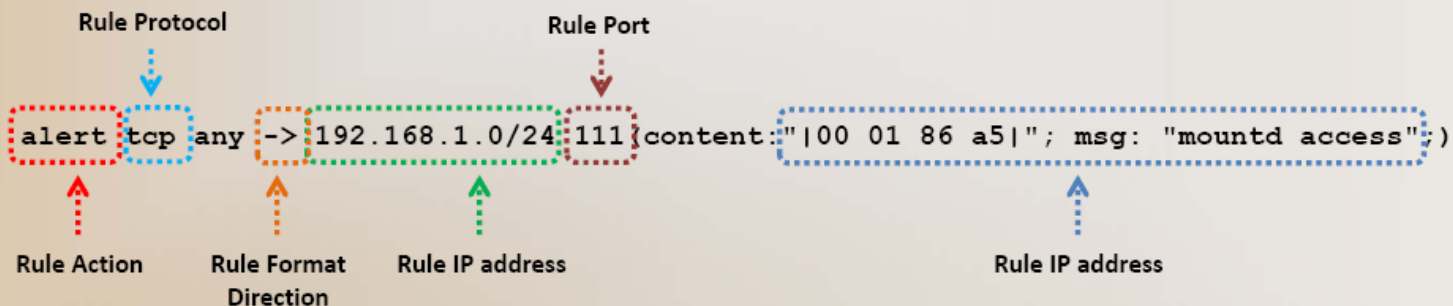
Administrator

- Decoder:** Saves the captured packets into heap, identifies link level protocols, and decodes IP
- Detection Engine:** It matches packets against rules previously charged into memory since Snort initialization
- Output Plug-ins:** These modules format the notifications for the user to access them in different ways (console, extern files, databases, etc)

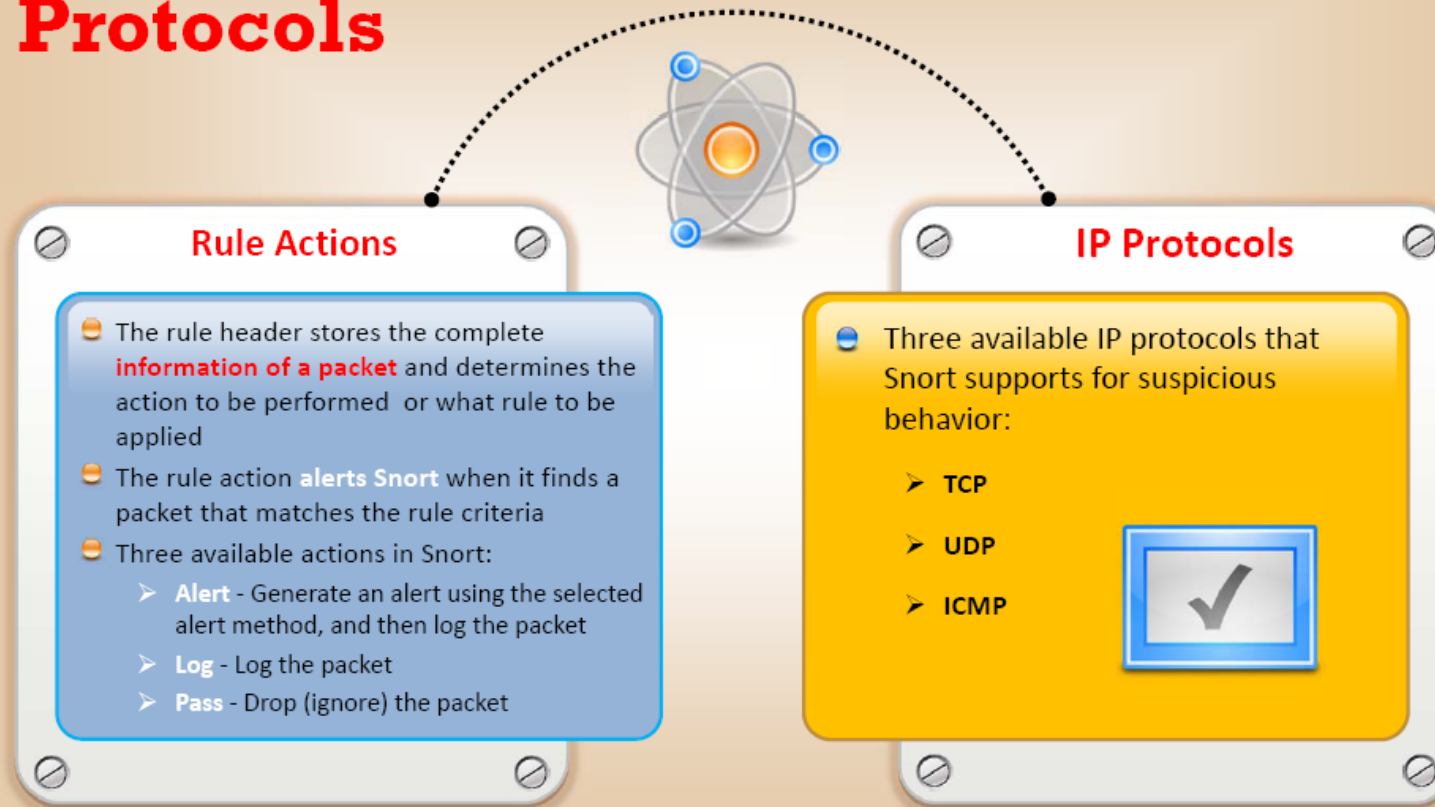
# Snort Rules

- Snort's rule engine enables one to write his/her **own rules** to meet the needs of the network
- Snort rules help in differentiating between **normal Internet activities** and **malicious activities**
- Snort rules must be contained on a **single line**, the Snort rule parser **doesn't handle rules on multiple lines**
- Snort rules come with two logical parts:
  - Rule header:** Identifies **rule's actions** such as alerts, log, pass, activate, dynamic, etc.
  - Rule options:** Identifies rule's **alert messages**

## Example:



# Snort Rules: Rule Actions and IP Protocols



# Snort Rules: The **Direction Operator** and **IP Addresses**

## The Direction Operator

- This operator indicates the **direction of the traffic** and the traffic can flow either in one direction (->) or bi-directionally (<>)
- Example Snort rules using the Bidirectional Operator:

```
log !192.168.1.0/24 any <> 192.168.1.0/24 23
```

## IP Addresses

- It deals with the **IP address and port information** for any particular rule
- Use keyword "**any**" to define any IP address
- Snort accepts addresses that are formed by a straight numeric IP address and a CIDR block applies netmask to the rule's address and to incoming packets that are verified against the rule
- Example IP Address Negation Rule:

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



# Snort Rules: **Port Numbers**

- Port numbers can be listed in different ways, including "any" ports, static port definitions, ranges, and by negation
- Port ranges are indicated with the range operator ":".
- Example of Port Negation



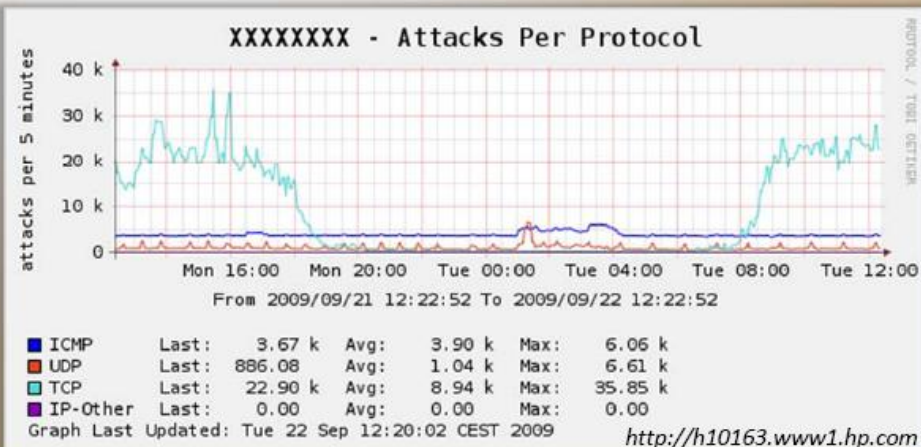
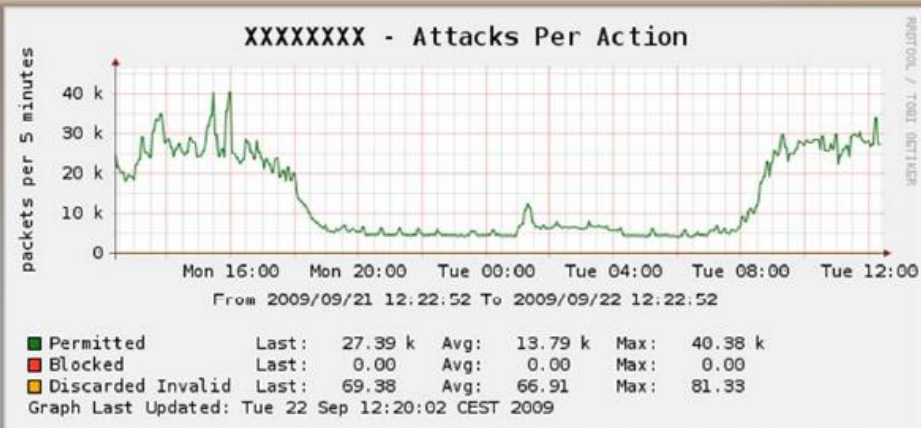
```
log tcp any any -> 192.168.1.0/24 !6000:6010
```

Protocols	IP address	Action
Log UDP any any ->	92.168.1.0/24 1:1024	Log UDP traffic coming from any port and destination ports ranging from 1 to 1024
Log TCP any any ->	192.168.1.0/24 :5000	Log TCP traffic from any port going to ports less than or equal to 5000
Log TCP any :1024 ->	192.168.1.0/24 400:	Log TCP traffic from privileged ports less than or equal to 1024 going to ports greater than or equal to 400

# Intrusion Detection System: **Tipping Point**



- TippingPoint IPS is inserted **seamlessly** and **transparently** into the network, it is an **in-line** device
- Each packet is thoroughly inspected to determine whether they are **malicious** or **legitimate**
- It provides **performance**, **application**, and **infrastructure** protection at gigabit speeds through total packet inspection



# Intrusion Detection Tools



**Security Network Intrusion Prevention System**  
<http://www-01.ibm.com>



**Strata Guard**  
<http://www.stillsecure.com>



**Peek & Spy**  
<http://networkingdynamics.com>



**CRCMd5 Data Validation**  
<http://www.forensics-intl.com>



**Cisco IDS 4250 Appliance Sensor**  
<http://www.cisco.com>



**DiskSearch 32**  
<http://www.forensics-intl.com>



**INTOUCH INSA-Network Security Agent**  
<http://www.ttinet.com>



**IDP8200**  
<http://www.juniper.net>



# Intrusion Detection Tools



**OSSEC**

<http://www.ossec.net>



**Netifera**

<http://netifera.com>



**eXpert-BSM**

<http://www.csl.sri.com>



**Cisco Intrusion Detection**

<http://www.cisco.com>



**AIDE (Advanced Intrusion Detection Environment)**

<http://aide.sourceforge.net>



**Tripwire**

<http://www.tripwire.com>



**SNARE (System iNtrusion Analysis & Reporting Environment)**

<http://www.intersectalliance.com>



**Vanguard Enforcer**

<http://www.go2vanguard.com>





# Firewall: Sunbelt Personal Firewall



# Firewalls



**Check Point Firewall Software Blade**

<http://www.checkpoint.com>



**eScan Enterprise**

<http://www.escan.com>



**Jetico Personal Firewall**

<http://www.jetico.com>



**ZoneAlarm Pro**

<http://www.zonealarm.com>



**Novell BorderManager**

<http://www.novell.com>



**FireWall-1**

<http://www.checkpoint.com>



**InstaGate**

<http://www.esoft.com>

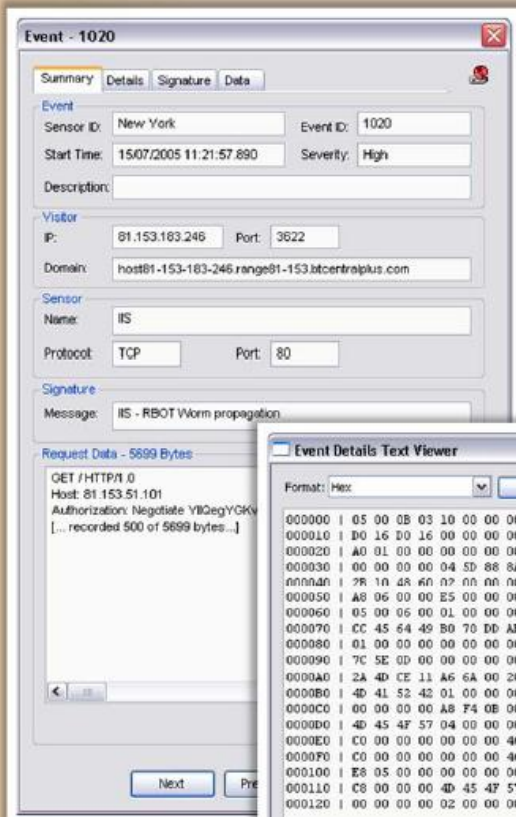


**AccessMaster NetWall**

<http://www.evidian.com>



# Honeypot Tool: **KFSensor**



Event - 1020

Summary Details Signature Data

Event

Sensor ID: New York Event ID: 1020

Start Time: 15/07/2005 11:21:57.890 Severity: High

Description:

Visitor

IP: 81.153.163.246 Port: 3622

Domain: host81-153-163-246.range81-153.btccentralplus.com

Sensor

Name: IIS

Protocol: TCP Port: 80

Signature

Message: IIS - RBOT Worm propagation

Request Data - 5699 Bytes

GET /HTTP/1.0

Host: 81.153.51.101

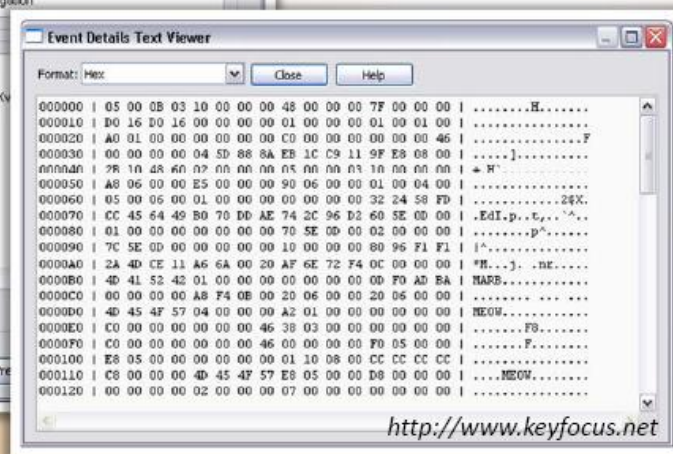
Authorization: Negotiate YWQegYCKV

[... recorded 500 of 5699 bytes ...]

Next Pre



- KFSensor is a **host-based** Intrusion Detection System (IDS)
- It acts as a honeypot to attract and detect hackers and worms by **simulating vulnerable system services** and **Trojans**

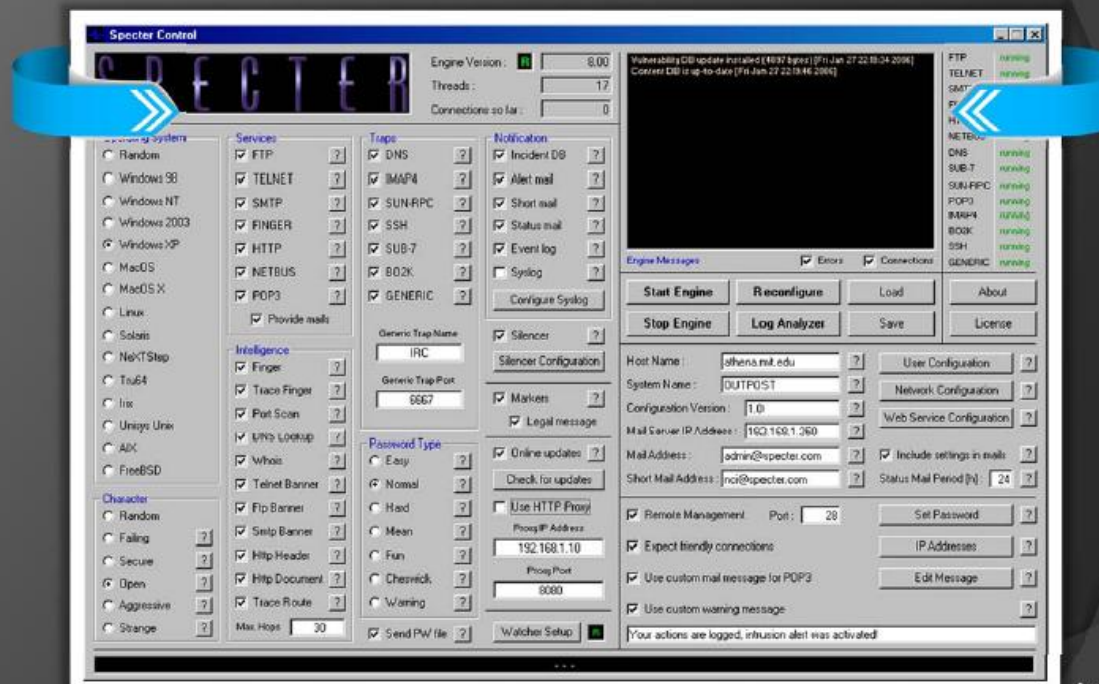


## Features:

- GUI based management console
- Remote management
- Snort compatible signature engine
- Emulations of Windows networking protocols
- Export logs in multiple formats
- Denial Of Service (DOS) attack protection

# Honeypot Tool: **SPECTER**

- SPECTER is a smart honeypot-based intrusion detection system that offers common **Internet services** such as SMTP, FTP, POP3, HTTP and TELNET which appear perfectly normal to the attackers but in fact are traps
- SPECTER provides massive amounts of **decoy content** including images, MP3 files, email messages, password files, documents and all kinds of software



<http://www.specter.com>





# Honeypot Tools



**NetBait**

<http://netbaitinc.com>



**Single-honeypot**

<http://single-honeypot.sourceforge.net>



**LaBrea Tarpit**

<http://labrea.sourceforge.net>



**Kojoney**

<http://kojoney.sourceforge.net>



**Sendmail SPAM Trap**

<http://www.tracking-hackers.com>



**HoneyBOT**

<http://www.atomicsoftwaresolutions.com>



**PatriotBox**

<http://www.alkasis.com>



**Google Hack Honeypot**

<http://ghh.sourceforge.net>

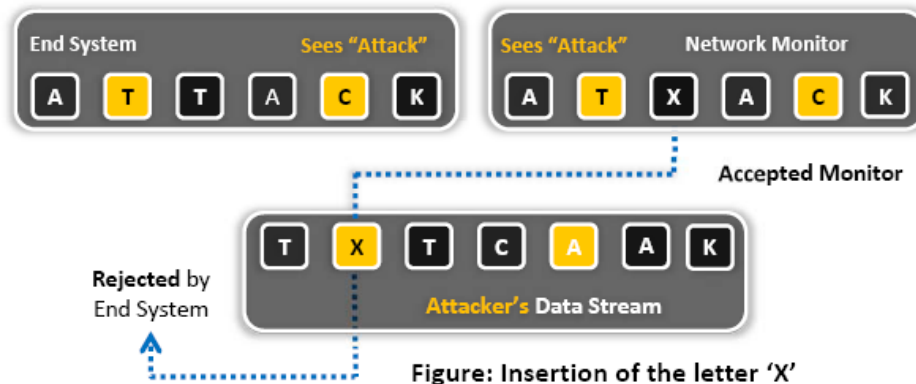


# Module Flow



# Insertion Attack

1. An IDS **blindly believes and accepts a packet** that an end-system has rejected
2. An attacker exploits this condition and **inserts data into the IDS**
3. This attack occurs when **NIDS is less strict in processing packets**
4. An attacker uses these attacks to **defeat signature analysis and sends request**, but obscures its contents on the IDS with additional data that make the request seem harmless
5. Here, the IDS gets more packets than the destination



- For example, the attacker can send packets whose Time to live fields have been crafted to reach the IDS but not the target computers
- An attacker confronts the IDS with a stream of 1-character packets (the attacker-originated data stream), in which one of the characters (the letter 'X') will be accepted only by the IDS
- As a result, the IDS and the end system reconstruct two different strings

# Evasion

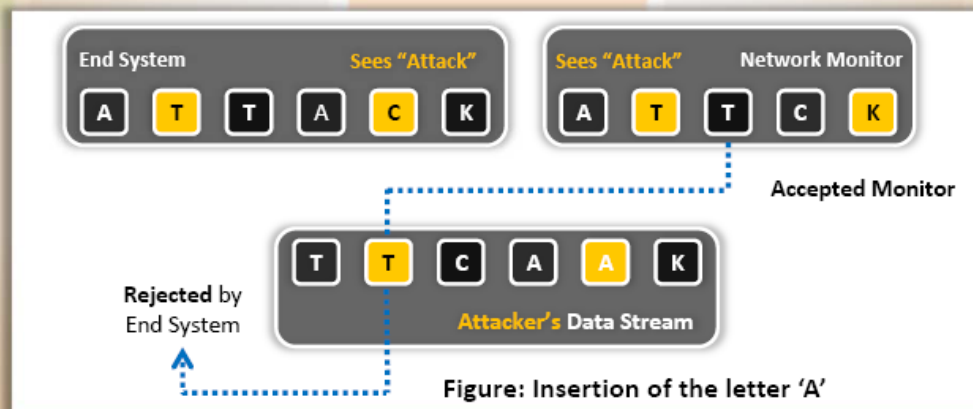
1 In this evasion technique, an end-system accepts a packet that an IDS rejects

2 Using this technique, an attacker exploits the host computer

3 Attacker sends portions of the request in packets that the IDS mistakenly rejects, allowing to remove parts of the stream from the ID system's view

4 Example, if the malicious sequence is sent byte-by-byte, and one byte is rejected by the IDS, the IDS cannot detect the attack

5 Here, the IDS gets fewer packets than the destination





# Denial-of-Service Attack (DoS)

- Many IDSs **employ central logging servers** that are used exclusively to **store IDS alert logs**
- The central server's function is to **centralize alert data** so it can be viewed as a whole rather than on a system-by-system basis
- However, if attackers know the central log server's IP address, they could **slow it down or even crash it using a DoS attack**
- After the server is shut down, **attacks could go unnoticed** because the alert data is no longer being logged



# Obfuscating



An IDS can be evaded by **obfuscating or encoding the attack payload** in a way that the target computer will reverse but the IDS will not

An attacker using the Unicode character could **encode attack packets that an IDS would not recognize** but that an IIS web server would decode and become attacked

**Polymorphic code** is another means to circumvent signature-based IDSs by creating unique attack patterns, so that the attack does not have a single detectable signature



An attacker **manipulates the path referenced in the signature** to fool the HIDS

**Attacks on encrypted protocols** such as HTTPS are obfuscated if the attack is encrypted



# False Positive Generation

- Another attack similar to the DoS method is to **generate a large amount of alert data** that must be logged
- Attackers craft packets known to trigger alerts within the IDS, forcing it to **generate a large number of false reports**
- This type of attack is designed to create a great deal of log "noise" in an attempt to **blend real attacks with the false**
- Attackers know all too well that when looking at log data, it can be very difficult to differentiate between legitimate attacks and false positives
- If attackers have knowledge of the IDS system, they can even **generate false positives specific to that IDS**





# Session Splicing

- 1 Session splicing is an IDS evasion technique that exploits how some IDSs do not reconstruct sessions before performing pattern matching on the data
- 2 The idea behind session splicing is to split data between several packets, making sure that no single packet matches any patterns within an IDS signature
- 3 if attackers know what IDS system is in use, they could add delays between packets to bypass reassembly checking
- 4 Many IDSs reassemble communication streams, so if a packet is not received within a reasonable amount of time, many IDSs stop reassembling and handling that stream
- 5 If the application under attack keeps a session active longer than an IDS will spend on reassembling it, the IDS will stop
- 6 As a result, any session after the IDS stops reassembling the sessions will be susceptible to malicious data theft by the attacker





# Unicode Evasion Technique



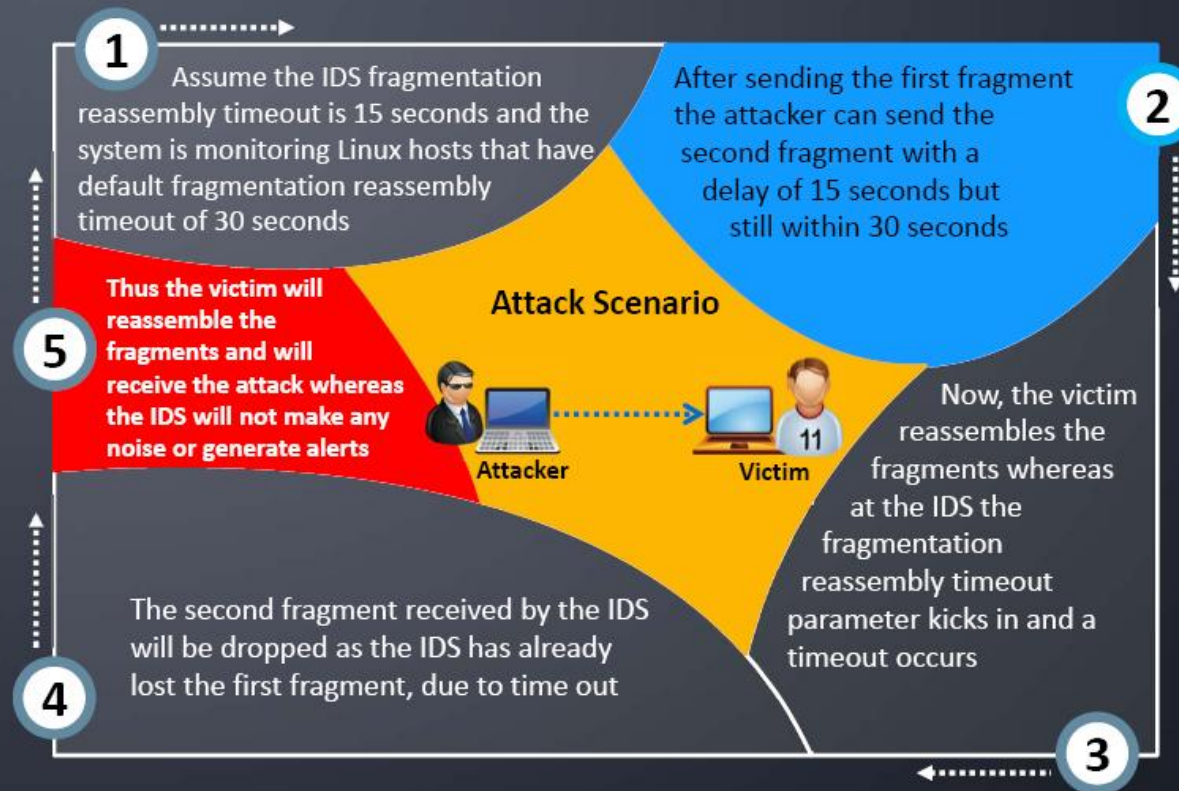
- Unicode is a **character representation** that gives each character a **unique identifier for each written language** to facilitate the uniform computer representation of each language
- This is problematic for IDS technology because it is possible to have **multiple representations of a single character**
- For example, '**\**' can be represented as **5C**, **C19C** and **E0819C**, which makes writing pattern matching signatures very difficult

## Example for how Unicode affects IDS:

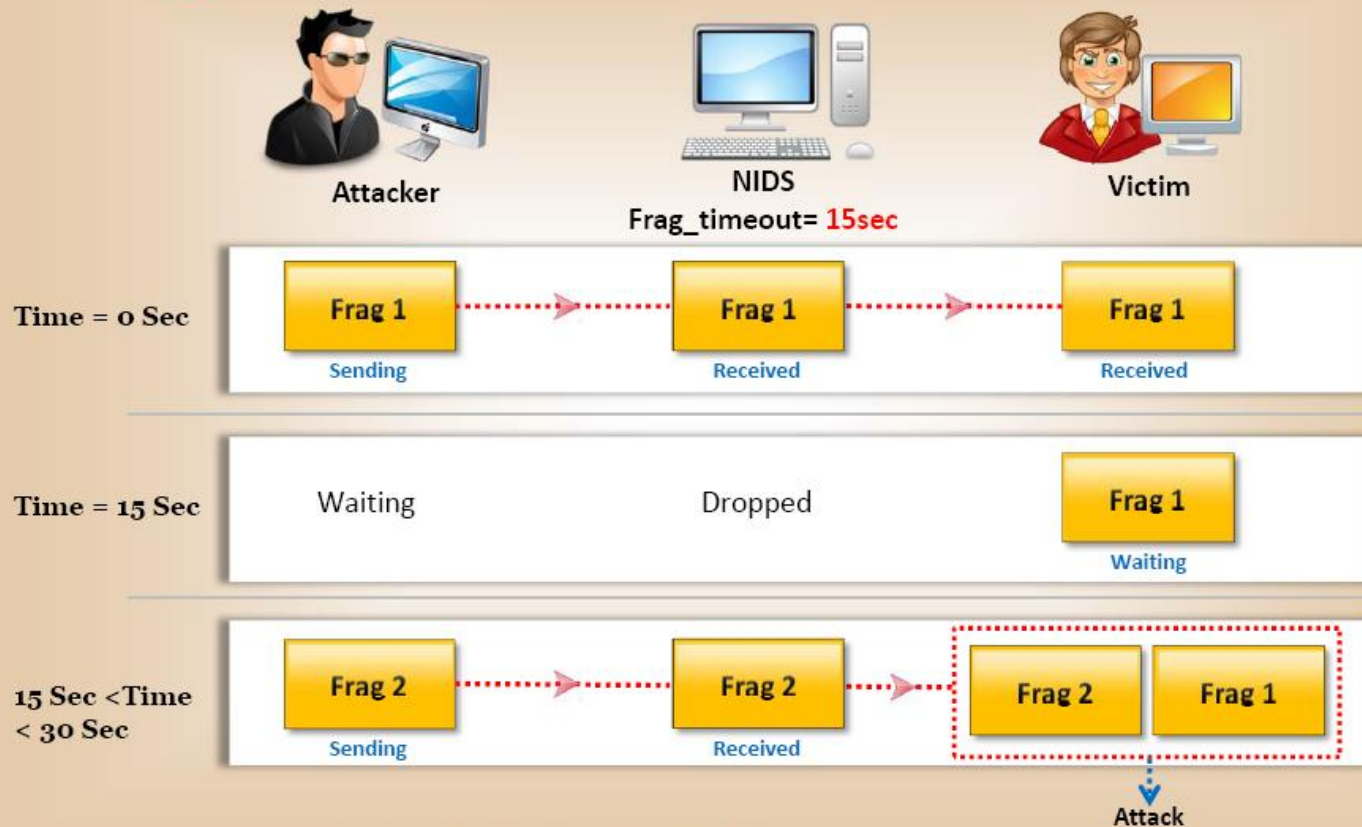
- Microsoft IIS 4.0/5.0 Directory Traversal vulnerability released in October 2000 by Rain Forrest Puppy
- This IIS vulnerability improperly **restricts directory listings** that were Unicode encoded within the URL request
- This allowed remote attackers to **view files on the IIS server** that they normally would not be permitted to see

# Fragmentation Attack

IDS fragmentation reassembly timeout is **less than** fragmentation reassembly timeout of the Victim

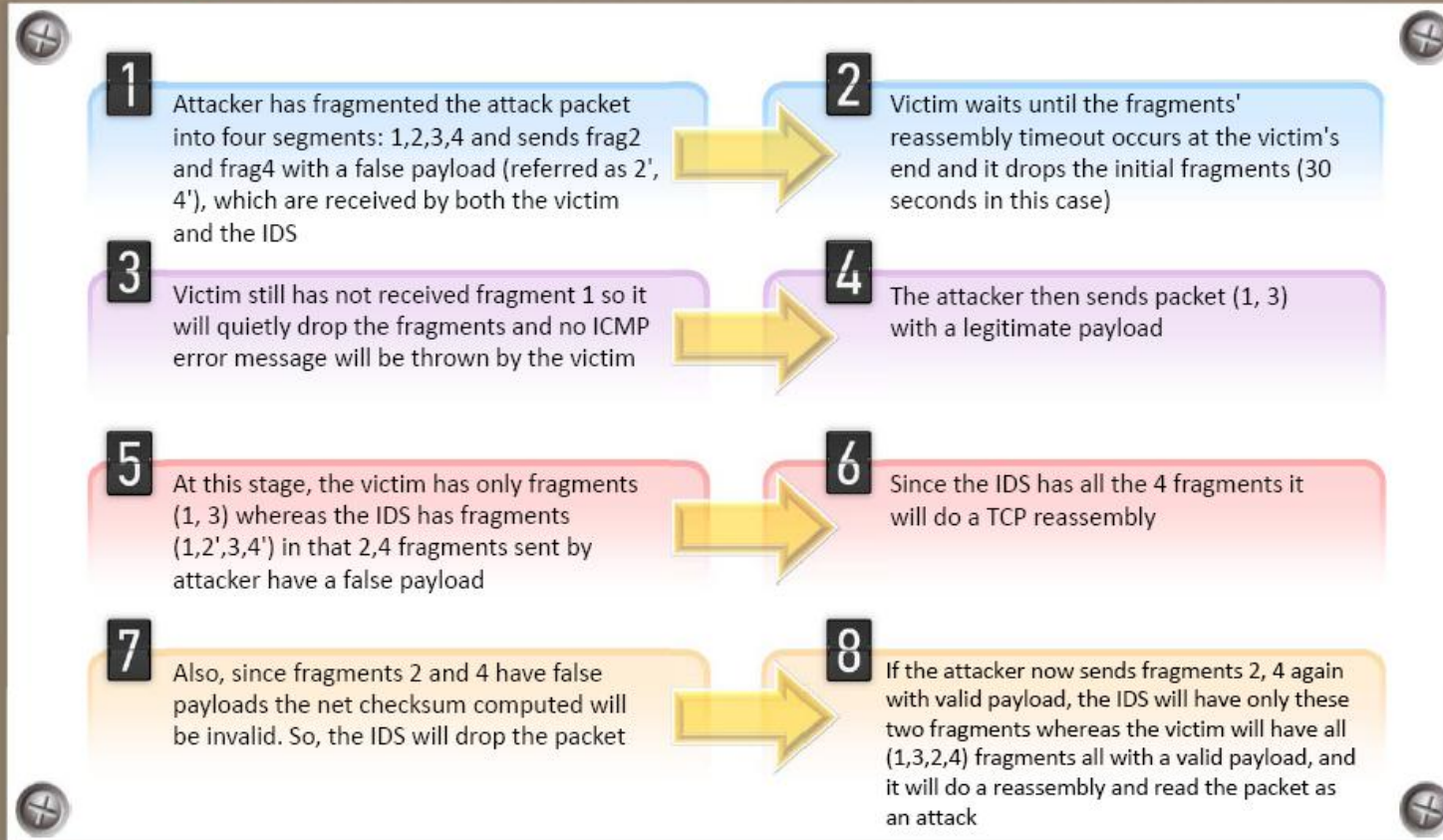


# Fragmentation Attack

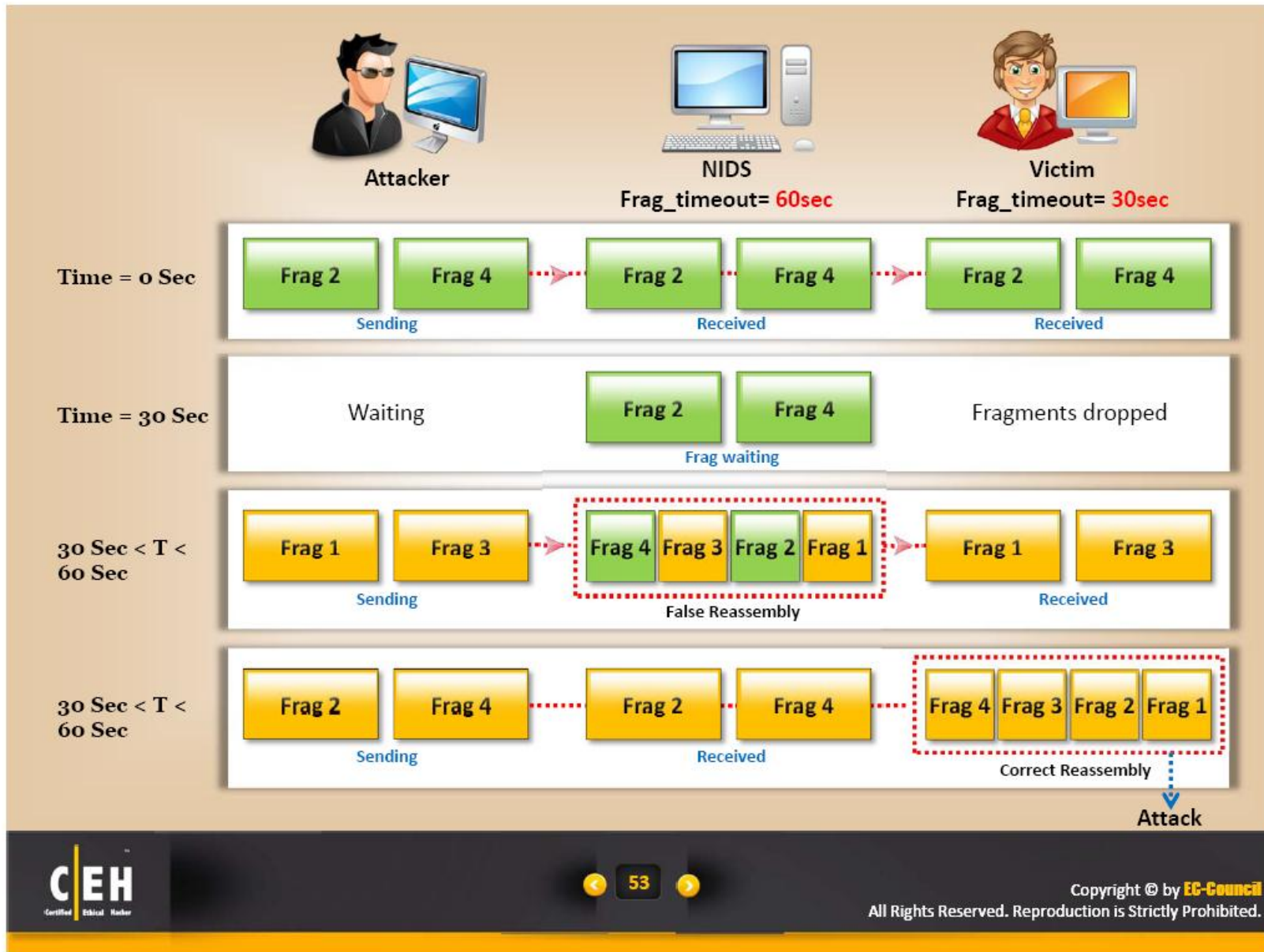


**Figure:** Attack where the NIDS fragmentation re-assembly timeout is less than the victim's fragmentation reassembly timeout

# Fragmentation Attack

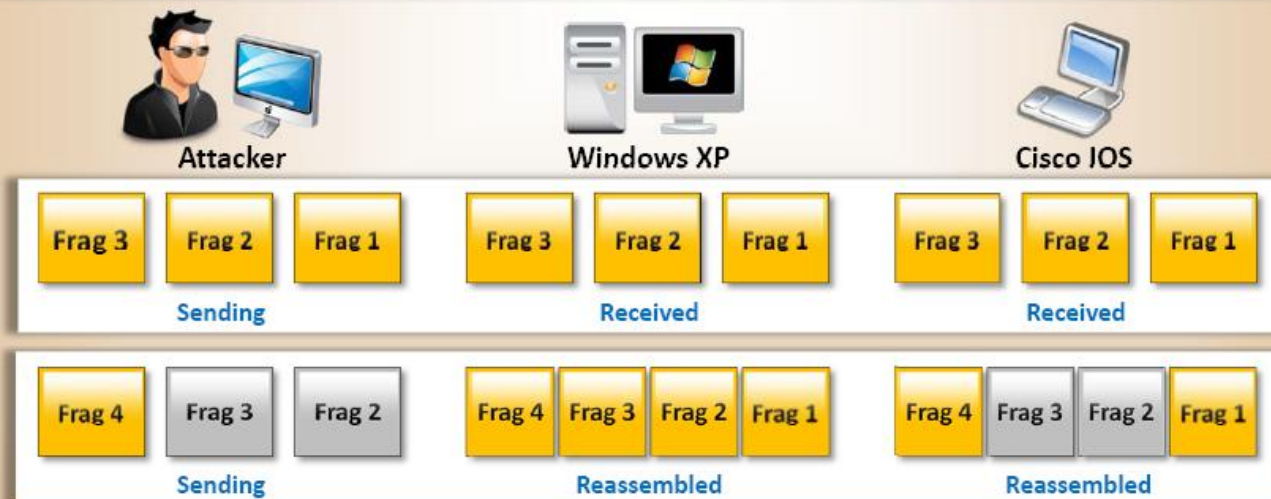






# Overlapping Fragments

- ✱ An IDS evasion technique is to **craft a series of packets** with TCP sequence numbers configured to overlap
- ✱ For example, the first packet will include **80 bytes** of payload but the second packet's sequence number will be **76 bytes** after the start of the first packet
- ✱ When the target computer **reassembles the TCP stream**, it must decide how to handle the four overlapping bytes
- ✱ Some operating Systems will take the **original fragments with a given offset** (For example, Windows W2K/XP/2003) and some operating Systems will take the subsequent fragments with a given offset (For example, Cisco IOS)



# Time-To-Live Attacks

- These attacks require the attacker to have a **prior knowledge of the topology** of the victim's network
- This information can be obtained by using tools such as **tracert** which give the information on the **number of routers between the attacker and the victim**

A router is present between the IDS and a victim - and the attacker is assumed to have this prior information and carries out the attack by **breaking it into three fragments**

Attacker sends fragment 1 with a **large TTL value** and this is received by both the IDS and the victim and then sends second fragment (frag2') with the TTL value of 1 and false payload

This fragment is received by the IDS whereas the router (which is situated between the IDS and the victim) discards it as the TTL value is now **reduced to zero**

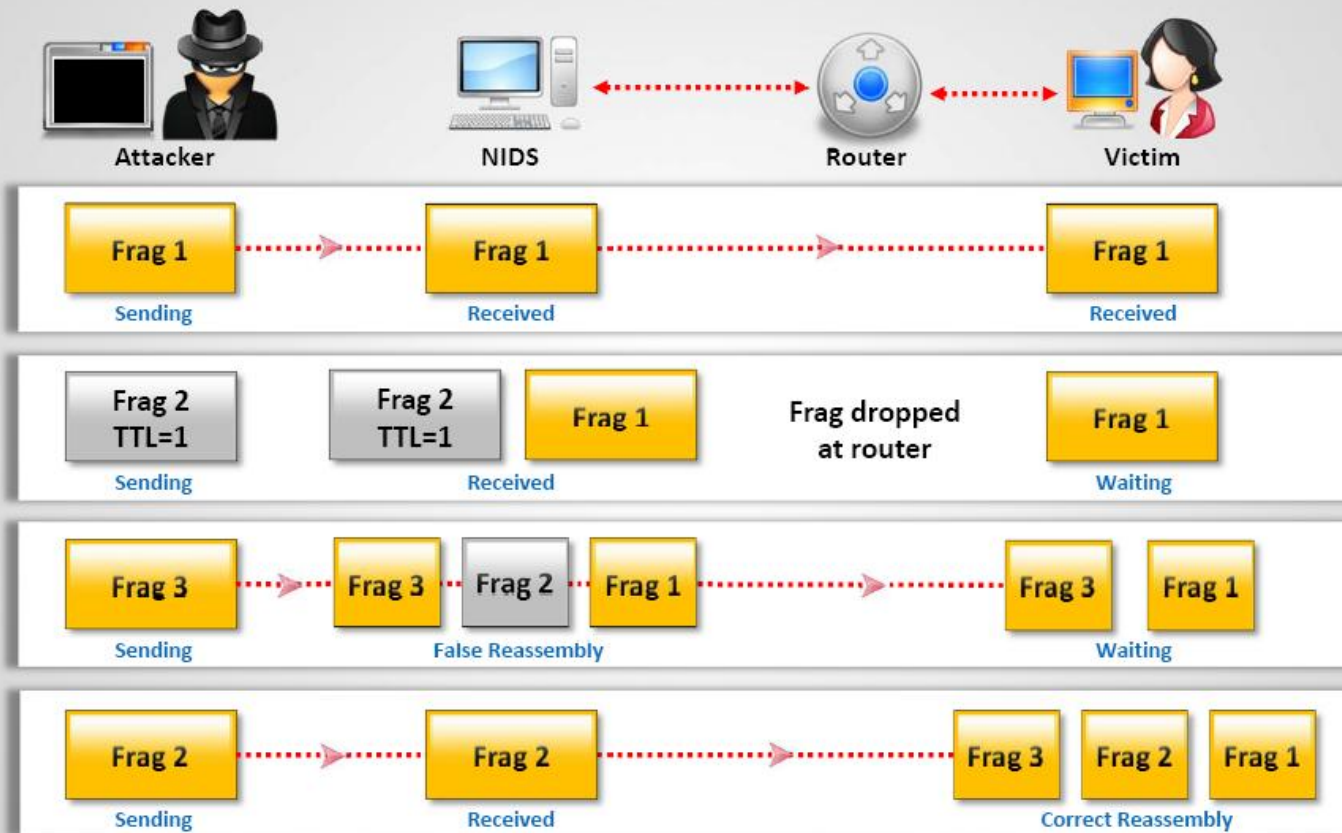
At this stage, the IDS has **only fragment 2** as it has already performed a reassembly and the stream has been flushed

The attacker finally sends the second fragment with a **valid payload** and the victim performs a reassembly on fragments (1, 2, 3) and gets the attack

The attacker then sends fragment 3 with a **valid TTL**. This makes the IDS perform a TCP-reassembly on fragments (1,2',3), whereas the victim still waits for the second fragment



# Time-To-Live Attacks



TTL based evasion attack



# Invalid RST Packets

1

The TCP protocol uses checksums to ensure that communication is reliable

2

A checksum is added to every transmitted segment and it is checked at the receiving end. When a checksum differs from the checksum expected by the receiving host, the packet is dropped at the receiver's end

3

The TCP protocol also uses an RST packet to end two-way communications

4

Attackers can use this feature to elude detection by sending RST packets with an invalid checksum, which causes the IDS to stop processing the stream because the IDS thinks the communication session has ended

5

However, the end host sees this packet and verifies the checksum value, then drops the packet if it is invalid

6

Some IDS systems might interpret this packet as an actual termination of the communication and stop reassembling the communication

7

Such instances allow attackers to continue to communicate with the end host while confusing the IDS because the end host accepts the packets that follow the RST packet with an invalid checksum value



# Urgency Flag



The urgency flag is used within the TCP protocol to mark data as urgent. TCP uses an urgency pointer that points to the beginning of urgent data within a packet

When the urgency flag is set, all data before the urgency pointer is ignored, and the data to which the urgency pointer points is processed



Attackers can place garbage data before the urgency pointer, and the IDS reads that data without consideration for the end host's urgency flag handling

This means the IDS has more data than the end host actually processed

Some IDSs do not take into account the TCP protocol's urgency feature, which could allow attackers to evade IDS, as seen in other evasion techniques



## Urgency flag attack example

"1 Byte data, next to Urgent data, will be lost, when Urgent data and normal data are combined."

Packet 1: ABC

Packet 2: DEF Urgency Pointer: 3

Packet 3: GHI

End result: ABCDEFHI

1. This example illustrates how the urgency flag works in conjunction with the urgency pointer
2. According to the 1122 RFC, the urgency pointer causes one byte of data next to the urgent data to be lost when urgent data is combined with normal data.

CEH  
Certified Ethical Hacker

58

Copyright © by EC-Council  
All Rights Reserved. Reproduction is Strictly Prohibited.



# Polymorphic Shellcode



# ASCII Shellcode



- ASCII shellcode contains only characters contained within the **ASCII standard**
- This form of shellcode allows attackers to **bypass commonly enforced character restrictions** within string input code
- It also helps attackers **bypass IDS pattern matching signatures** because strings are hidden within the shellcode in a similar fashion to polymorphic shellcode
- Using ASCII for shellcode is very **restrictive** in that it limits what the shellcode can do under some circumstances because not all assembly instructions convert directly to ASCII values
- This restriction can be bypassed using **other instructions or a combination of instructions** that convert to ASCII character representation, which serves the same purpose of the instructions that improperly convert

The following is an ASCII shellcode example:

```
char shellcode[] =  
"LLLLYhb0pLX5b0pLHSSPPWQPPaPWSUTBRDJfh5  
tDS"  
"RajYX0Dka0TkafhN9fyf1Lkb0TkdfY0Lkf0Tk  
gfh"  
"6rfYf1Lki0tkkh95h8Y1LkmjpY0Lkq0tkrh2wn  
uX1"  
"Dks0tkwjfX0Dkx0tkx0tkyCjnY0LkzC0TkzCCj  
tX0"  
"DkzC0tkzCj3X0Dkz0TkzC0tkzChjG3IY1LkzCC  
CC0"  
"tkzChpfCMX1DkzCCCC0tkzCh4pCnY1Lkz1TkzC  
CCC"  
"fhJGfXf1Dkzf1tkzCCjHX0DkzCCCCjvY0LkzCC  
Cjd"  
"X0DkzC0TkzCjWX0Dkz0TkzCjdX0DkzCjXY0Lkz  
0tk"  
"zMdgvvn9F1r8F55h8pG9wnuvjrNfrVx2LGkG3I  
Dpf"  
"cM2KgmnJGgbinYshdvD9d";
```

When executed, the shellcode above executes a **"/bin/sh"** shell. 'bin' and 'sh' are contained in the last few bytes of the shellcode.



# Application-Layer Attacks

1

Many applications that deal with media such as images, video and audio employ some form of compression to be sent in a form much smaller than the original which increases data transfer speeds



2

When a flaw is found in these applications, the entire attack can occur within compressed data, and the IDS will have no way to check the compressed file format for signatures



3

Many IDSs look for specific conditions that allow for an attack. However, there are times when the attack can take many different forms



4

For example, integer overflow vulnerabilities could be exploited using several different integer values



5

This fact combined with compressed data makes signature detection extremely difficult



# Desynchronization – Pre Connection SYN



This attack calls **bind** to get the kernel to assign a local port to the socket before calling **connect**

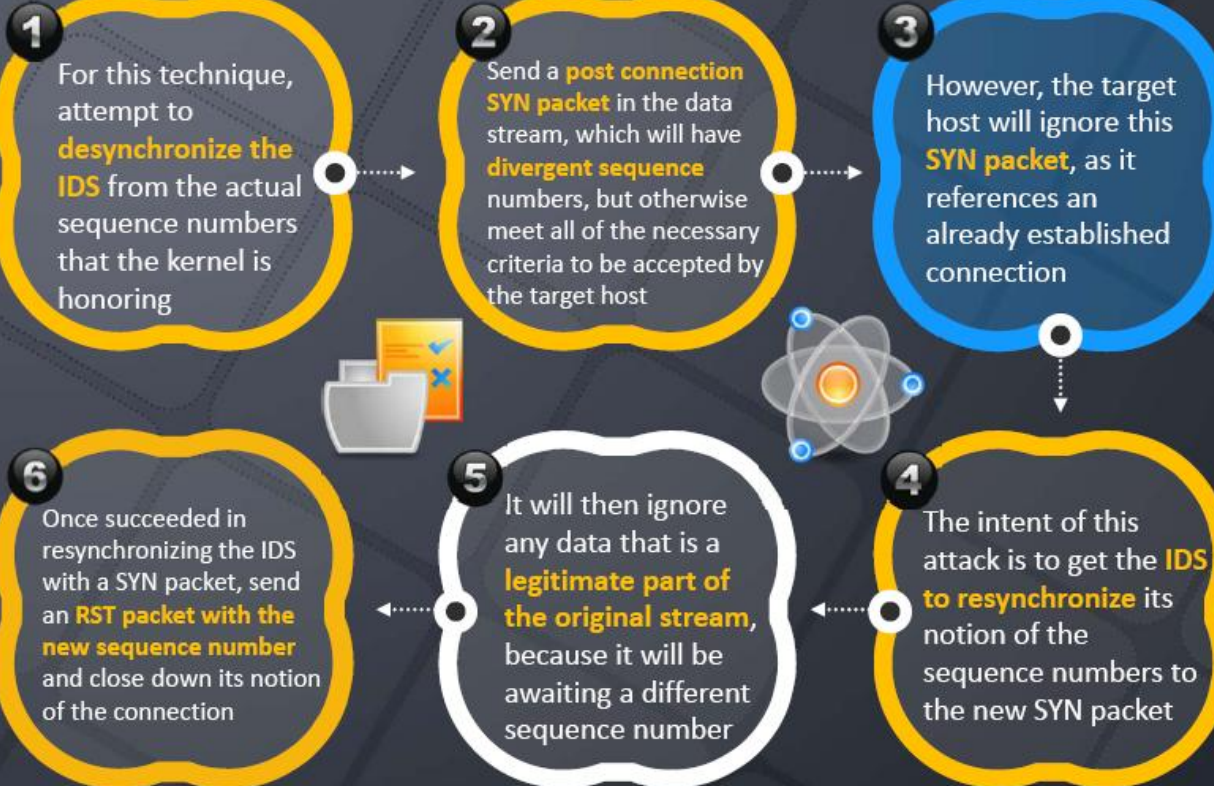


Send an **initial SYN** before the real connection, with an invalid TCP checksum



If the sniffer ignores subsequent SYNs in a connection, and does not check the **TCP checksum**, then this attack will **synchronize the sniffer/IDS** to a bogus sequence number before the real connection occurs

# Desynchronization - Post Connection SYN



# Other Types of **Evasion**

## Encryption

If the attacker already establishes an **encrypted session with the victim**, this results to be the most effective evasion attack

The attacker sends loads of **unnecessary traffic to produce noise** and if the IDS does not analyze the noise traffic, the true attack traffic may go undetected

## Flooding





# Module Flow

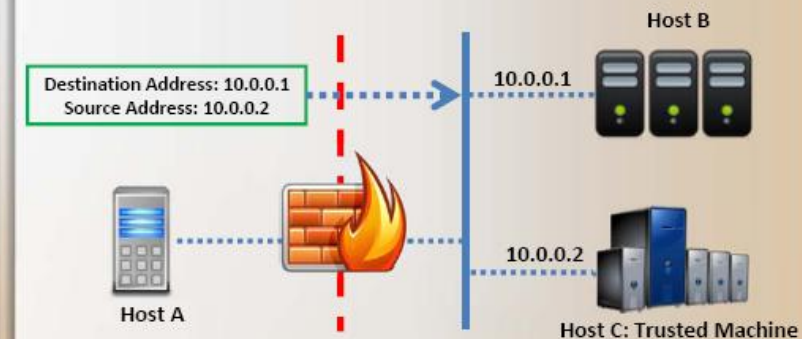


# IP Address Spoofing

- Using this technique, the attacker gains an **unauthorized access** to a computer or a network by making it appear that the message comes from a trusted machine by **"spoofing" the IP address** of that machine
- To bypass the firewall, the attacker **modifies the address information** in IP packet header and the source address bits field



- For example, let's consider **three hosts** A, B and C
- Host **C is a trusted machine** of host B
- Host A wants to send some packets to host B and **A impersonates itself to be C** by changing the IP address of these packets
- When these packets are received, **B thinks that these packets are from C**, but actually they are from A

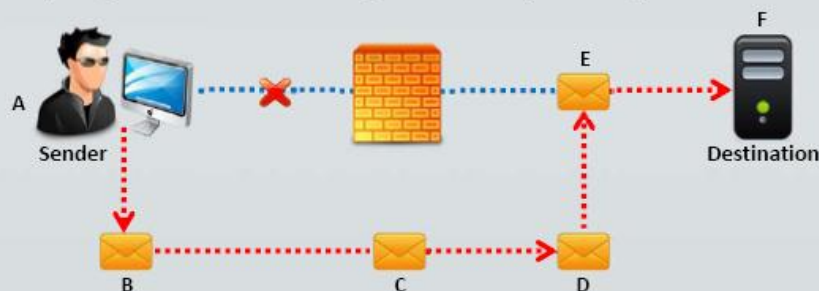


# Attacking Session Token Generation Mechanism

- Using this technique, the **sender of the packet** designates the route that a packet should take through the network
- When these packets travel through the nodes in the network, each router will check **IP address of the destination** and choose the next node to forward them
- In source routing, the **sender** makes some or all of these decisions on the router



The figure shows the principle of the source routing but it is an optimal way, which makes the decision of the next hop



# Tiny Fragments

- The attacker uses the IP fragmentation technique to create **extremely small fragments** and **force the TCP header information into the next fragment**
- This may result in a case whereby the TCP flags field is forced into the second fragment, filters will be **unable to check these flags in the first octet** thus ignoring them in subsequent fragments
- Attackers hope that only the **first fragment is examined** by the filtering router and the remaining fragments are passed through
- This attack is used to **avoid user defined filtering rules** and works when the **firewall checks only for the TCP header information**



IP-3ar0JI0B0K				MK=1, Fragment Offset=0			
Source Port			Destination Port				
Sequence Number							
Acknowledgement Sequence Number							
Data Offset	Reserved	-	ACK	-	-	-	Window
Checksum							Urgent Pointer=0
0							





# Bypass Blocked Sites Using **IP Address** in Place of **URL**

This method involves typing the **IP address** directly in browser's address bar in place of typing the blocked website's domain name



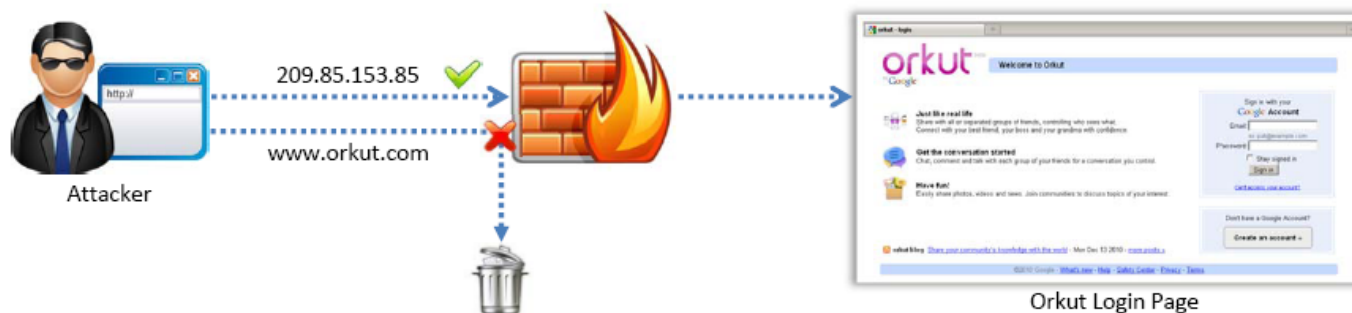
For example instead of typing **www.Orkut.com**, type its **IP address** to access Orkut



**Host2ip** can help you to **find the IP address** of that blocked website



If the blocking software can track the IP address sent to the web server the **website could not be unblocked** or accessed by using this method



# Bypass Blocked Sites Using Anonymous Website Surfing Sites

- Many websites around the net enable to **surf the internet** anonymously.
- Some websites provide options to **encrypt the URL's** of the websites
- These proxy websites will **hide actual IP address** and will show another IP address, this could **prevent the website being blocked** thus allowing to access them



Here is a list of some of the **proxy servers** that can help you to **unblock the blocked websites**:

- <http://www.anonymizer.com>
- <http://anonymouse.org>
- <http://proxify.com>
- <http://www.bumsk.com>
- <http://www.dailybestlinks.com>
- <http://www.spysurfing.com>
- <http://alienproxy.com>
- <http://indianproxy.com>

## Bypass a Firewall using **Proxy Server**



1. Find a appropriate proxy server



2. On the Tools menu of any Internet browser, go to LAN of Network Connections tab, and then click LAN/Network Settings



3. Under Proxy server settings select the use a proxy server for LAN



4. In the Address box, type the IP address of the proxy server

5. In the Port box, type the port number that is used by the proxy server for client connections (by default, 8080)



6. Click to select the Bypass proxy server for local addresses check box if you do not want the proxy server computer to be used when connected to a computer on the local network

7. Click OK to close the LAN Settings dialog box

8. Click OK again to close the Internet Options dialog box

# Bypassing Firewall through **ICMP Tunneling** Method



It allows to tunnel a backdoor shell in the **data portion of ICMP Echo packets**

**RFC 792**, which **delineates ICMP operation**, does not define what should go in the data portion

The **payload portion is arbitrary** and is not examined by most of the firewalls, thus any data can be inserted in the payload portion of the ICMP packet, including a backdoor application

Some administrators keep **ICMP open on their firewall** because it is useful for tools like **ping** and **traceroute**

Assuming that ICMP is allowed through a firewall, use **Loki ICMP tunneling** to execute commands of choice by **tunneling them inside the payload of ICMP echo packets**



Attacker

Wraps evil client command in **ICMP Echo packet**



Firewall

Unwraps command, **executes it locally wraps output** in ICMP Echo packet and resends back to attacker



Internet Client



# Bypassing Firewall through **ACK Tunneling** Method



It allows to tunnel backdoor application with **TCP packets with the ACK bit set**

ACK bit is used to **acknowledge receipt of a packet**

Some firewalls **do not check packets with the ACK bit set** because ACK bits are supposed to be used in response to legitimate traffic that is already being allowed through

Tools such as **AckCmd** (<http://ntsecurity.nu>) can be used to implement ACK tunneling



Attacker

Wraps evil client command in **TCP packet**



Firewall

Unwraps command, **executes it locally wraps output** in TCP packet and resends back to attacker



Internet Client

# Bypassing Firewall through **HTTP Tunneling** Method



This method can be implemented if the target company has a **public web server** with **port 80 used for HTTP traffic**, that is unfiltered on its firewall

Many firewalls **do not examine the payload of an HTTP packet** to confirm that it is legitimate HTTP traffic, thus it is possible to tunnel traffic inside TCP port 80 because it is already allowed

Tools such as **HTTPtunnel** (<http://www.nocrew.org>) use this technique of tunneling traffic **across TCP port 80**

HTTPtunnel is a **client/server application**, the client application is called **htc** and the server is **hts**

Upload the **server onto the target system** and tell it which port is to be redirected through **TCP port 80**



Attacker

Wraps evil client command in **payload of HTTP packet**



Firewall

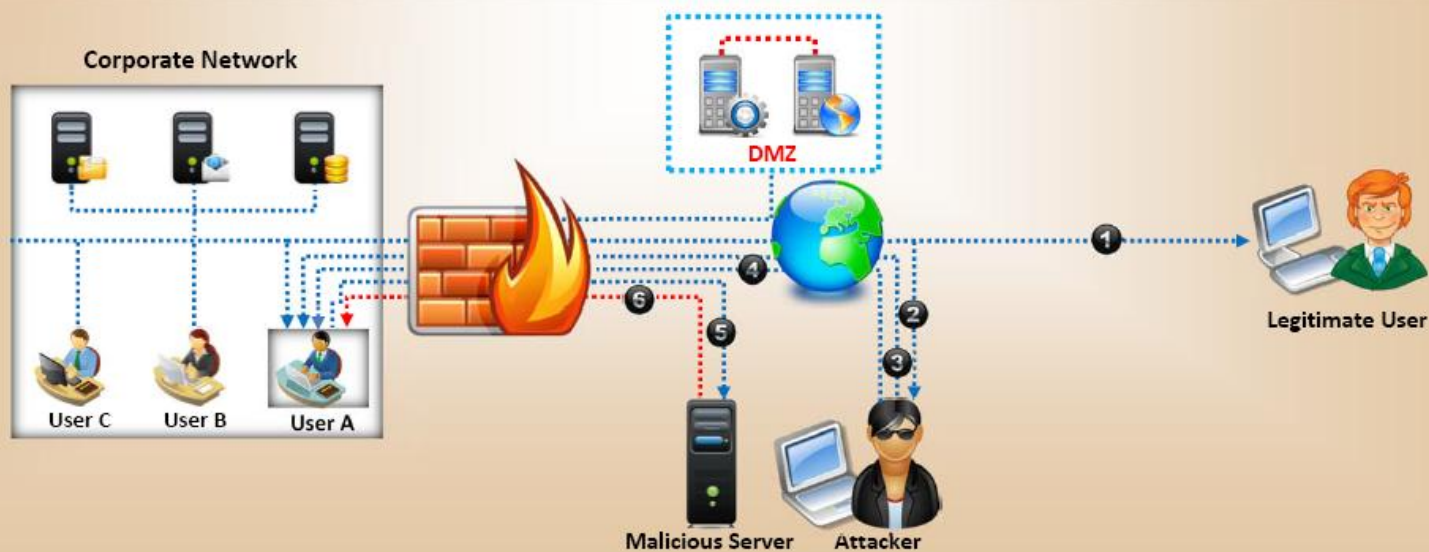
Unwraps command, **executes it** locally wraps output in **payload of HTTP packet** and resends back to attacker



Internet Client

# Bypassing Firewall through **External Systems**

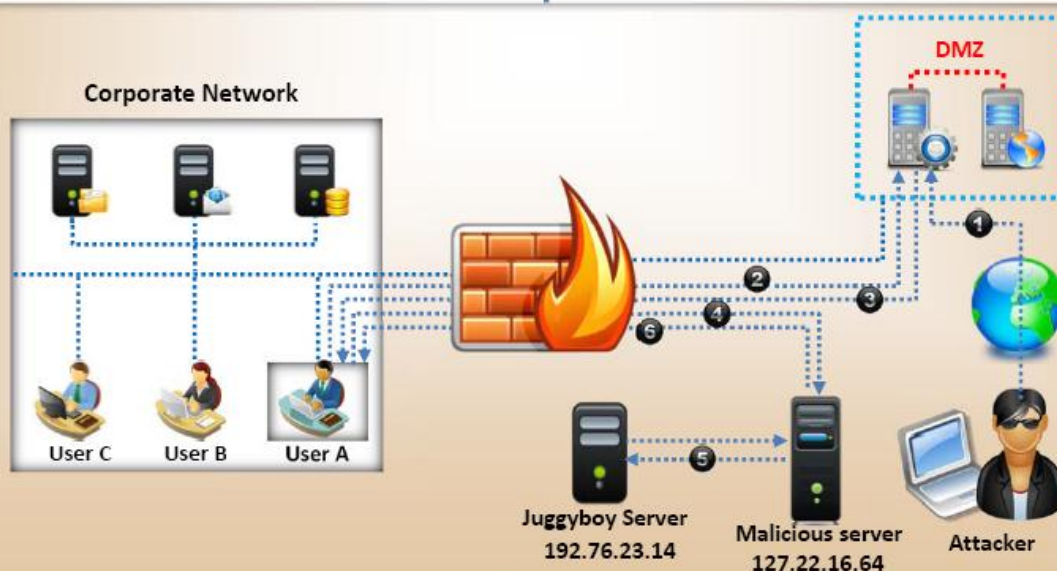
1. Legitimate user works with some **external system** to access the corporate network
2. Attacker sniffs the **user traffic**, steals the **session ID** and **cookies**
3. Attacker **accesses the corporate network** bypassing the firewall and gets **Windows ID** of the running Netscape 4.x/ Mozilla process on user's system
4. Attacker then issues an **openURL() command** to the found window
5. User's web browser connects with the **attacker's WWW server**
6. Attacker inserts **malicious payload into the requested web page** (Java applet) and thus attacker's **code gets executed** on the user's machine



# Bypassing Firewall through MITM Attack



1. Attacker performs **DNS server poisoning**
2. User A requests for WWW.juggyboy.com to the **corporate DNS server**
3. Corporate DNS server sends the **IP address (127.22.16.64) of the attacker**
4. User A accesses the **attacker's malicious server**
5. Attacker connects with the **real host and tunnels the user's HTTP traffic**
6. Attacker inserts **malicious payload into the requested web page** (Java applet), and thus **attacker's code gets executed** on the user's machine





# Module Flow



# Detecting Honeypots

Attackers can determine the presence of honeypots by probing the services running on the system

Attackers craft malicious probe packets to scan for services such as HTTP over SSL (HTTPS), SMTP over SSL (SMTPS), and IMAP over SSL (IMAPS)

Some of the tools that can be used to probe honeypots include:

- Send-safe Honeypot
- Hunter
- Nessus
- Hping

Ports that show a particular service running but deny a three-way handshake connection indicate the presence of a honeypot

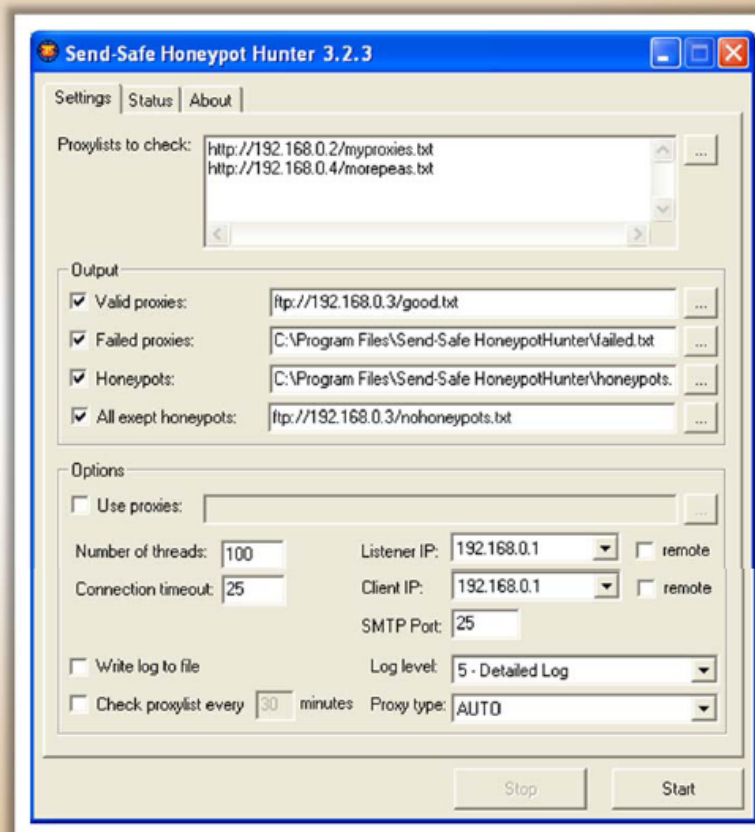
**Note:** Attackers can also defeat the purpose of honeypots by using multi-proxies (TORs) and hiding their conversation using encryption and steganography techniques

# Honeypot Detecting Tool: **Send-Safe HoneyPot Hunter**

Send-Safe HoneyPot Hunter is a tool designed for checking **lists of HTTPS and SOCKS proxies** for "honey pots"

## Features:

- Checks lists of **HTTPS, SOCKS4 and SOCKS5 proxies** with any ports
- Checks **several remote or local proxylists** at once
- Can **upload "Valid proxies"** and **"All except honeypots"** files to FTP
- Can process **proxylists** automatically every specified period of time
- May be used for **usual proxylist** validating as well



<http://www.send-safe.com>



# Module Flow



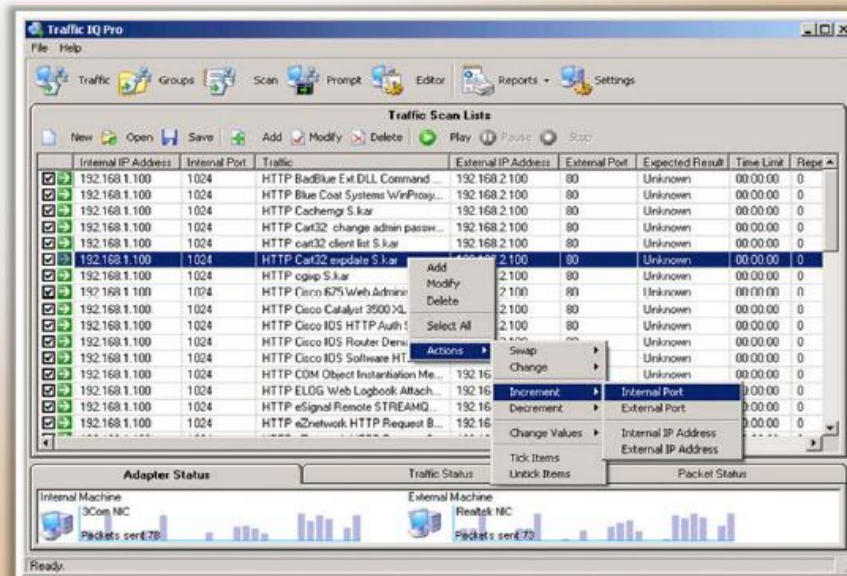


# Firewall Evasion Tool: **Traffic IQ Professional**

- Traffic IQ Professional enables security professionals to **audit and validate the behavior of security devices** by generating the **standard application traffic or attack** traffic between two virtual machines

Traffic IQ Professional can be used to **assess, audit, and test the behavioral characteristics** of any non-proxy packet-filtering device including:

- Application layer firewalls
- Intrusion detection systems
- Intrusion prevention systems
- Routers and switches



<http://www.blade-software.com>



# Firewall Evasion Tool: **tcp-over-dns**

tcp-over-dns contains a special **dns server** and a special **dns client**. The client and server work in tandem to provide a **TCP** (and UDP!) **tunnel** through the standard DNS protocol

```
C:\Utilities\tcp-over-dns-1.0>java -jar tcp-over-dns-server.jar --domain dnstun... --forward-port 22
000000.0 main: tcp-over-dns-server starting up
000000.0 main: Hosting domain: 
000000.0 main: DNS listening on: /0.0.0.0:53
000000.0 main: Forwarding to: /127.0.0.1:22
000000.0 main: MTU: 1500
000000.0 main: Log level: 3
```

<http://analogbit.com>

# Firewall Evasion Tools



## Covert TCP

<http://www.firstmonday.dk>



## AckCmd

<http://ntsecurity.nu>



## Tomahawk

<http://tomahawk.sourceforge.net>



## Traffic IQ Gateway

<http://www.karalon.com>



## Snare Agent

<http://www.intersectalliance.com>



## Your Freedom

<http://www.your-freedom.net>



## TCPOpera

<http://www.csif.cs.ucdavis.edu>



## Atelier Web Firewall Tester

<http://www.atelierweb.com>



# Packet Fragment Generators



**Blast**

<http://www.foundstone.com>



**MGEN Toolset**

<http://cs.itd.nrl.navy.mil>



**Ettercap**

<http://ettercap.sourceforge.net>



**Net::RawIP**

<http://search.cpan.org>



**hping2**

<http://www.hping.org>



**SING**

<http://sourceforge.net>



**Libnet**

<http://libnet.sourceforge.net>



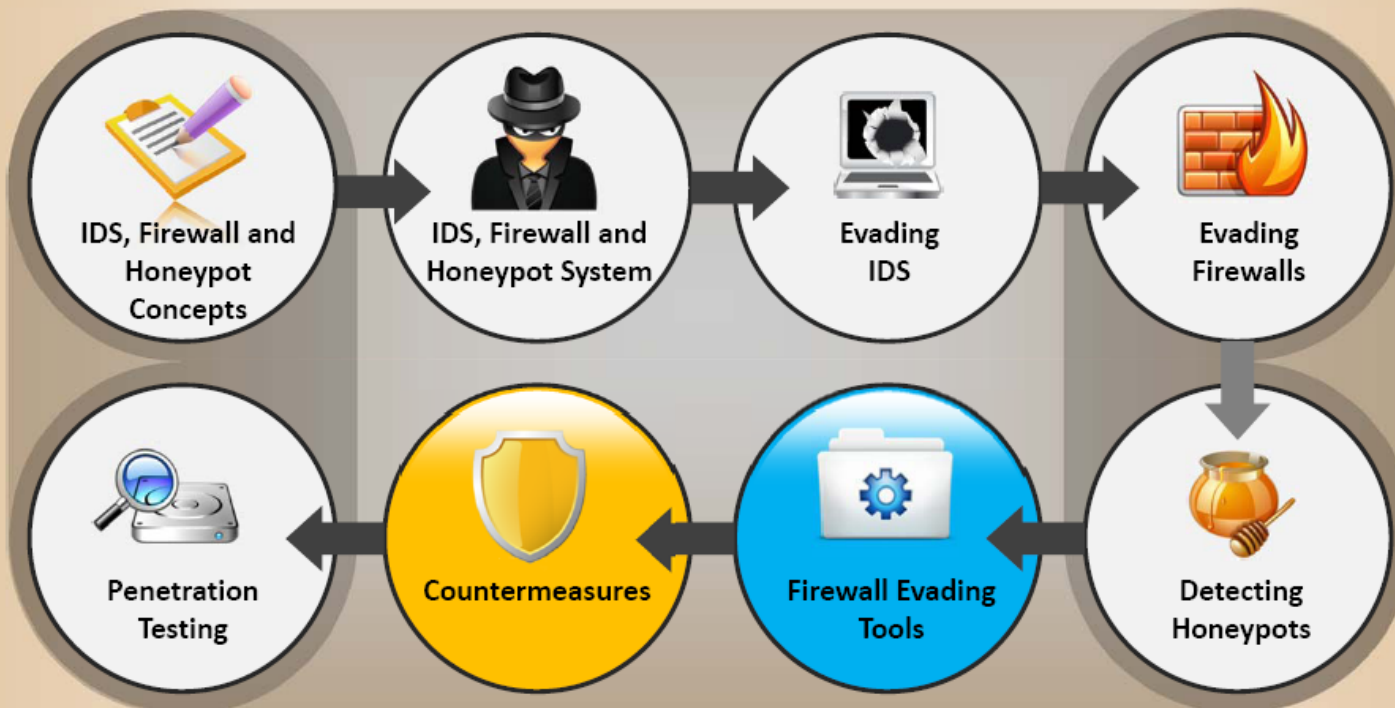
**Nconvert**

<http://www.xnview.com>





# Module Flow



# Countermeasures

Administratively **shut down a switch port interface** associated with a system from which attacks are being launched



Look for the **nop opcode** other than 0x90 to defend against the polymorphic shellcode problem

Perform "**bifurcating analysis**", in which the monitor deals with ambiguous traffic streams by instantiating separate analysis threads for each possible interpretation of the ambiguous traffic



Maintain **security vulnerability awareness, patch vulnerabilities** as soon as possible and wisely choose the IDS based on the network topology and network traffic received

Generate **TCP RST packets to tear down malicious TCP sessions**, or issues any of several available **ICMP error code packets** in response to malicious UDP traffic



Interact with the **external firewall or router** to add a general rule to block all communication from individual IP addresses or entire networks

# Countermeasures



Implement a **"traffic normalizer"**: a network forwarding element that attempts to **eliminate ambiguous network traffic** and reduce the amount of connection state that the monitor must maintain

Ensure that IDSs **normalize fragmented packets** and allow those packets to be reassembled in the proper order, which enables the IDS to look at the information just as the end host will see it



Keep **updating** the IDS system and firewall software regularly



Maintain **security vulnerability awareness, patch vulnerabilities** as soon as possible and wisely choose the IDS based on the **network topology and network traffic** received



Change the **TTL field to a large value**, ensuring that the end host always receives the packets. In such case attackers cannot slip information to the IDS. As a result, that **data never reaches the end host**, leaving the end host with the malicious payload

# Module Flow





# Firewall/IDS Penetration Testing

Firewall/IDS penetration testing is to evaluate the security by **testing for ingress and egress vulnerabilities** and **proper rule sets of the entire network** with respect to the possibility of entry from an external location



To check if firewall/IDS properly enforces an **organization's** firewall/IDS policy

To check if firewall/IDS and components within network properly enforce an **organization's** network security policy

How well does the firewall/IDS provide protection **against** externally initiated attacks

To check how effective is the **network's** security perimeter

## Why Firewall/IDS pen testing?



To check how much **information** about a network is available from outside a network

To check the firewall/IDS for **potential breaches of security** that can be exploited

To evaluate the **correspondence of** firewall/IDS rules with respect to the actions performed by them

To verify whether the **security policy** is correctly enforced by a sequence of firewall/IDS rules or not



START.....

# Firewall Penetration Testing

Footprint  
the target

Perform Port Scanning  
to detect firewall

Firewall  
detected?

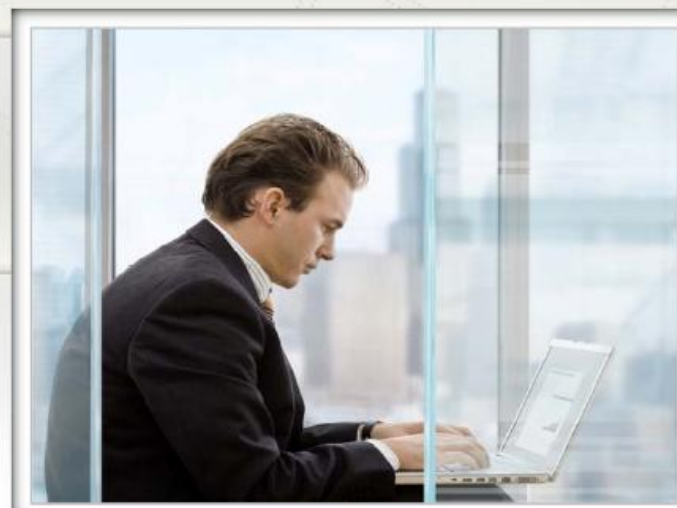
Perform Banner  
Grabbing to detect  
firewall

Firewall  
detected?

Perform  
Firewalking to  
detect firewall

Firewall  
detected?

No firewall



- Perform port scanning technique to know the **available ports** that uniquely identifies the firewalls
- Perform banner grabbing technique to detect the **services run** by the firewall
- Perform firewalking technique to determine **access information** on the firewall, when probe packets are sent

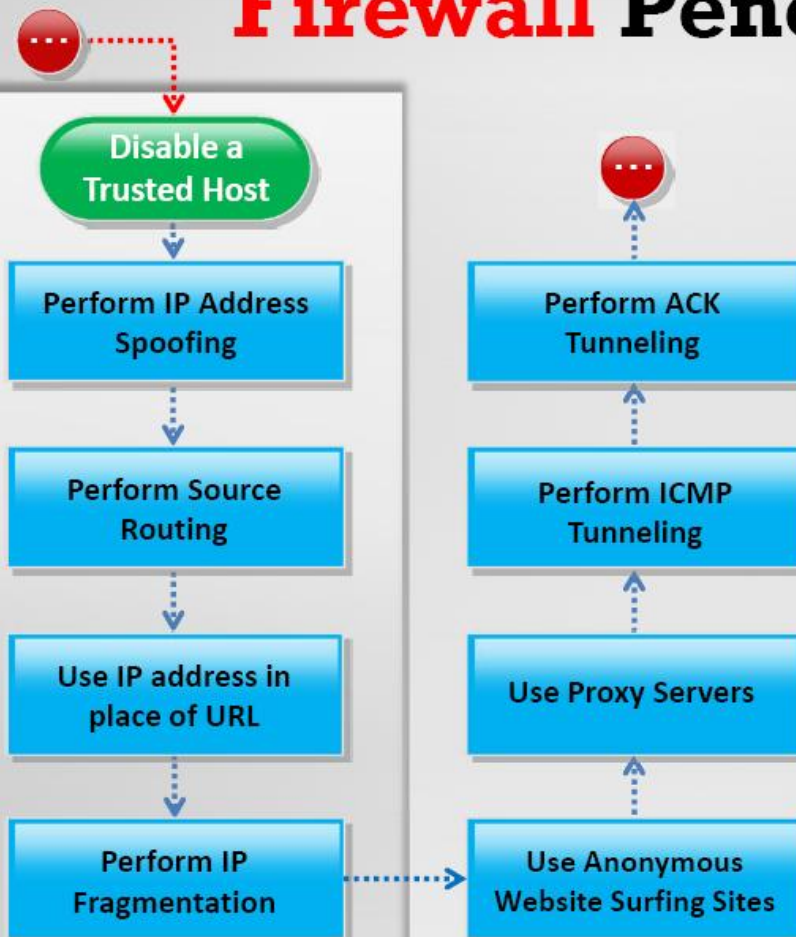
CEH  
Certified Ethical Hacker

90

Copyright © by EC-Council  
All Rights Reserved. Reproduction is Strictly Prohibited.



# Firewall Penetration Testing



- Perform **IP address spoofing** to gain unauthorized access to a computer or a network
- Perform **fragmentation attack** to force the TCP header information into the next fragment in order to bypass the firewall
- Use **proxy servers** that block the actual IP address and display another thereby allowing to **access the blocked website**
- Perform **ICMP tunneling** to tunnel a backdoor application in the data portion of ICMP Echo packets
- Perform **ACK tunneling** using tools such as **AckCmd** to tunnel backdoor application with TCP packets with the ACK bit set

# Firewall Penetration Testing



Perform HTTP Tunneling

Use External Systems

Perform MITM Attack

Document All the Findings



- Perform HTTP tunneling using tools such as **HTTPTunnel** to tunnel the traffic across TCP port 80
- Gain **access to the corporate network** by sniffing the user's traffic and stealing the session ID and cookies
- Perform M-I-T-M attack in order to **own corporate DNS server** or to spoof DNS replies to it

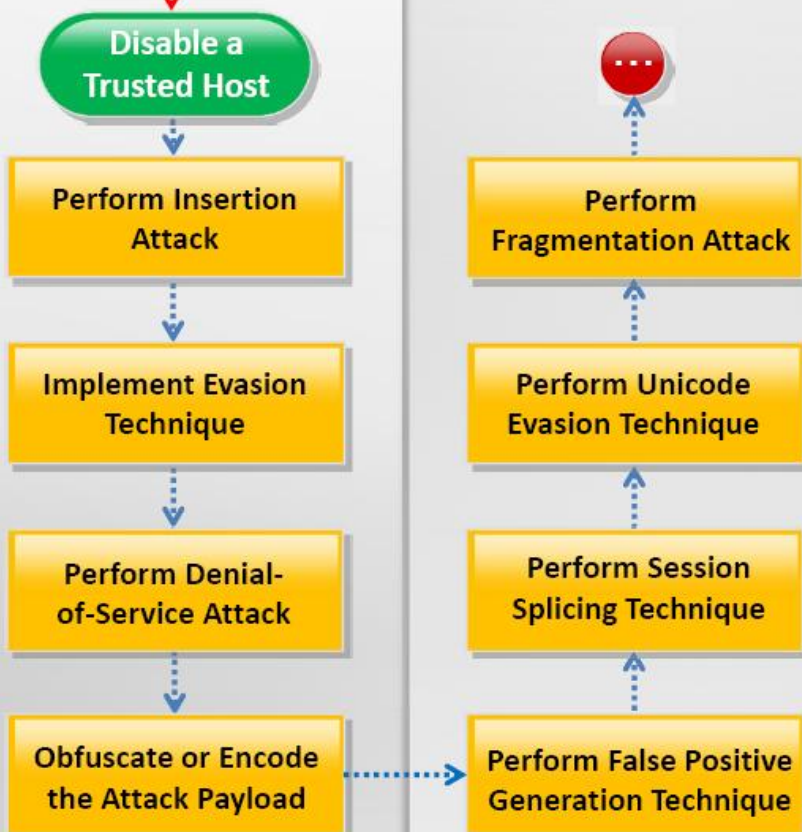




# IDS Penetration Testing



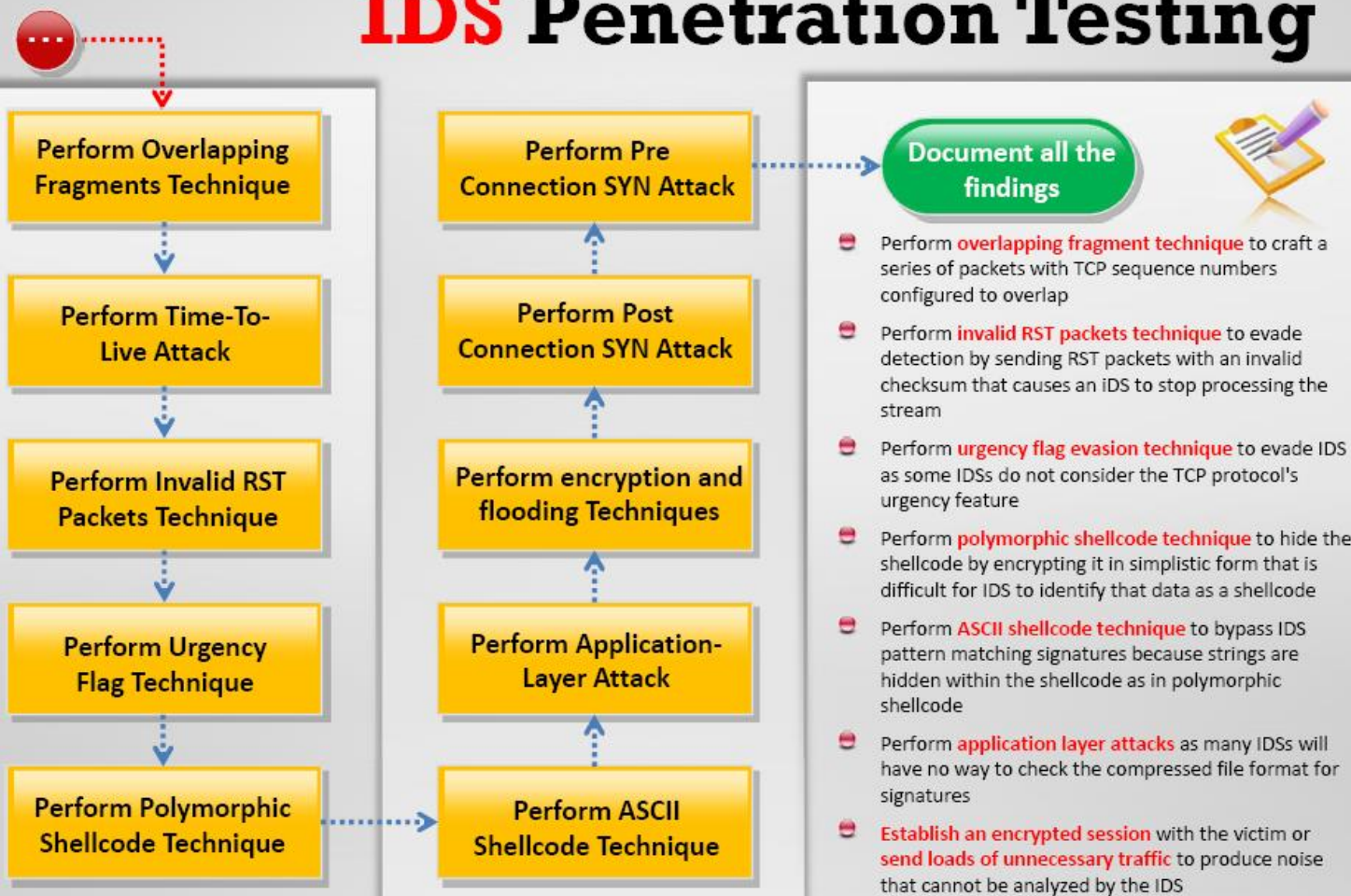
START



- Perform **obfuscating technique** to encode attack packets that IDS would not detect but an IIS web server would decode and become attacked
- Perform **false positive generation technique** to create a great deal of log "noise" in an attempt to blend real attacks with the false
- Perform **session splicing technique** to stop the IDS by keeping the session active for longer time than IDS will spend on reassembling it
- Perform **Unicode evasion technique** to evade IDS as it is possible to have multiple representations of a single character
- Perform **fragmentation attack with** IDS fragmentation reassembly timeout **less and more than** that of the Victim



# IDS Penetration Testing



# Module Summary

- ☐ Intrusion Detection Systems (IDS) monitor packets on the network wire and attempt to discover if an attacker is trying to break into a system
- ☐ System Integrity Verifiers (SIV) monitor the system files to find when an intruder changes. Tripwire is one of the popular SIVs
- ☐ Intrusion detection happens either by anomaly detection or signature recognition or Protocol Anomaly Detection
- ☐ Firewall is a hardware or software or combination of both designed to prevent unauthorized access to or from a private network
- ☐ Firewall is identified by three techniques namely port scanning, banner grabbing, and firewalking
- ☐ Honeypots are programs that simulate one or more network services that are designated on a computer's ports
- ☐ In order to effectively detect intrusions that use invalid protocol behavior, IDS must re-implement a wide variety of application-layer protocols to detect suspicious or invalid behavior
- ☐ One of the easiest and most common ways for an attacker to slip by a firewall is by installing network software on an internal system, that uses a port address permitted by the firewall's configuration



# Quotes

“As we've come to realize, the idea that security starts and ends with the purchase of a prepackaged firewall is simply misguided.”

- **Art Wittmann**,  
Managing Director,  
Informationweek Analytics

