Global Information Assurance Certification Paper

GIAC Practical Assignment

Intrusion Detection In-Depth

GCIA Version 3.0

SANSFIRE 2001

July 30 ~ August 4

Author: Chih-yu Lin

Date: December 2001

Table of Contents

Assignment 1: Describe the State of Intrusion Detection: Overview of Cisco Secure IDS Signatures

Introduction

Cisco Secure Intrusion Detection Systems employ a signature-based intrusion detection technology to detect the misuse of network resources. They scan network packets for known attack signatures, and then take user-defined action upon detecting an attack. This document introduces Cisco Secure IDS signature architecture, as well as how to configure signatures on Cisco Secure IDS sensor and Cisco Secure Policy Manager (CSPM).

In the first part of the document, we will provide an overview of Cisco Secure IDS Signature Architecture. Then we will introduce Signature configuration methods on CSPM and Cisco Secure IDS sensor[1]. Additionally, we will provide the Cisco Secure IDS Signature tuning recommendation.

Signature Architecture

Cisco Secure Intrusion Detection Systems use signature-based intrusion detection technology. A signature is a pattern or a collection of information that can be used to detect a common attack. This section will introduce two important components of Cisco Secure IDS Signature: Signature Engines and Signature Parameters. By configuring Signature Engines and Signature Parameters, we can create new signatures and perform signatures customization for existing signatures.

Signature Engine

A signature engine is a component of the Cisco IDS Sensor designed to support many signatures in a certain category. An engine is composed of a parser and an inspector. Each engine has a set of legal parameters, which have allowable ranges or sets of values. When creating custom signatures, it is important to choose the correct engine. Signature Engines are described as following:

ATOMIC.* Engines

ATOMIC Engines are used to create simple, single packet conditions that cause alarms to be triggered. Each packet condition has specialized parameters that deal with each of the protocol-specific inspections within the scope of the engine.

FLOOD.* Engines

FLOOD Engines are used to create signatures that watch for any host sending multiple packets to a single host.

[1] This document is based on CSPM v2.3.2i, Cisco Secure IDS Sensor v3.0

SWEEP.* Engines

SWEEP.PORT Engines are used to create signatures that deal with port connections between two nodes. Signatures can be designed to detect an attacker probing for available services of a specific host.

SWEEP.RPC Engine is similar to the port sweep but only counts valid RPC requests made to a set of ports. Signatures can be designed to detect attackers probing for an RPC service available on a specific host.

SWEEP.HOST Engine is used to create signatures that deal with host sweeps from a single node to many nodes. Signatures can be designed to detect probes for a single service across multiple hosts or more obscure network discovery techniques.

SERVICE.* Engines

SERVICE engines are used to create signatures that deal with the Layer 5+ protocol of the service. The SERVICE.DNS.TCP/UDP engines support analysis of compressed messages and can fire alarms on request/reply conditions and overflows. The SERVICE.RPC and SERVICE.PORTMAP engines are fine tuned for RPC and Portmapper requests. Batch and fragmented messages are decoded and analyzed.

STRING.* Engines

STRING engines are used to create signatures that search for REGEX strings in a stream of traffic.

Signature Parameters

Signature Parameters define the specific detection criteria for each signature. The two categories of Signature Parameters are described below:

Master Signature Parameters:

Master Signature Parameters define general signature properties that can be applied to all Signature Engines. For example:

- MaxInspectLength parameter defines the maximum number of bytes to inspect.
- WantFrag parameter indicates whether a fragment is desired.

Engine-Specific Parameters:

Every Signatures Engine contains both Master Signature Parameters and Engine-Specific Parameters. The Engine-Specific Parameters define specific properties for each Signature Engine. For example:

- IcmpId parameter defines ICMP header identifier value on ATOMIC.ICMP Signature Engine.
- *Rate* parameter defines maximum allowed packets per second on FLOOD.HOST.UDP Signature Engine.

For the complete list of Signature Parameters, please refer to Cisco Intrusion Detection System Signature Engine version 3.0:
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids6/13346_01.htm

Signature Configuration on CSPM

On CSPM, the attack signatures used by the sensor are stored in a logical grouping called a sensor signature.

The Sensor Signatures branch on CSPM is where you create, store, and modify the sensor signatures used to configure sensors.

Sensor signatures define the types of intrusions, or attacks, that you want the sensor to detect and report. They also define any additional actions that you want the sensor to take upon detecting a particular attack.

We will first introduce signature categories on CSPM, and then go into signatures configuration detail.

Signature Category

Within a sensor signature, individual attack signatures are further subdivided into four categories: Embedded, Connection, String-matching and ACL signatures.

Embedded Signatures

Embedded (general) signatures are known attack signatures that are included in the sensor software. The list of embedded attack signatures available to a sensor depends upon the version of software the sensor is running. If you are using sensor software version 3.0 or later, you can add embedded signatures.

Connection Signatures

Connection signatures are user-configurable attack signatures based on the protocol (TCP or UDP) and port number of the traffic being monitored.

String-matching Signatures

String-matching signatures are user-configurable attack signatures based on data carried by packets. String-matching signatures use regular expressions to perform string matching on the packet data payload. Furthermore, string-matching signatures can be configured to examine only incoming, outgoing, or bi-directional network traffic for the string.

ACL Signatures

ACL signatures are user-configurable attack signatures based on policy violations recorded by network devices in the syslog stream. To configure the sensor to detect ACL signatures, you must first configure one or more routers to log ACL violations. Then, you must configure the router to communicate with the sensor, and configure the sensor to accept syslog traffic from the router.

Signature Configuration

Creating Sensor Signature

Sensor signatures can be created directly under the Sensor Signatures branch of the Tools and Services tree: (Figure 1)

- Expand the Tools and Services tree and the Sensor Signatures branch
- Right-click the **Sensor Signatures** branch icon in the Navigator pane, point to New, and then click **Sensor Signature** on the shortcut menu.
- Type the name in the selected Name box, and then press **Enter**

By default, the new sensor signature has all general signatures enabled, set to low, medium, or high severity, and the Actions set to None; only selected connection signatures enabled, set to low or medium severity, and the Actions set to None; and all pre-defined string signatures enabled with the Actions set to None. The new sensor signature has no default ACL signatures defined.

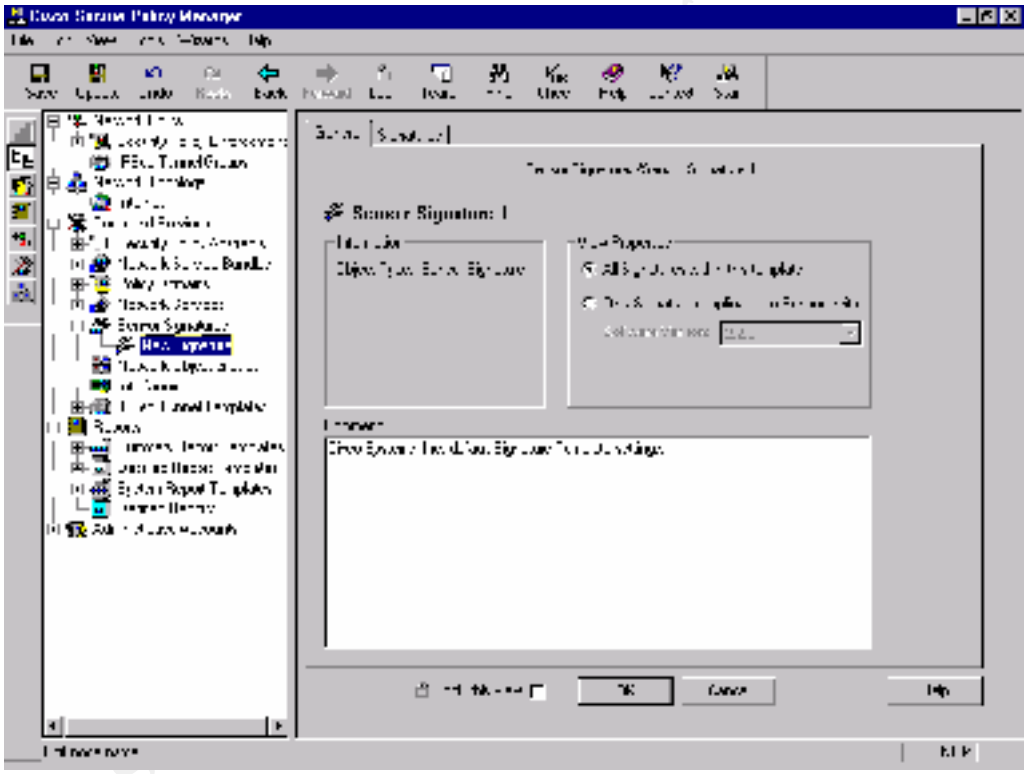

Figure 1 – Sensor Signature branch on CSPM

Configuring General Signatures

You can configure the following properties for each attack signature listed in the General Signatures panel: (Figure 2)

- Severity: Categorizes the attack signature. The severity setting is used in Event Viewer to distinguish among the types of attacks being logged. Severity levels: **Low**, **Medium and High.**

- Enable: Configures the sensor to scan network traffic for that particular attack signature. The sensor generates an alarm when it detects the attack. Disabling an attack signature causes the sensor to disregard any network traffic that displays the signature.
- Actions: Determines what action the sensor will take when it detects an attack signature. Select one or more of the following actions for each attack signature:
  - ♦ Block: The sensor issues a command to a PIX Firewall or a Cisco router. That device then denies the traffic from attacker host or network
  - ♦ TCP Reset: The sensor resets the TCP session in which the attack signature was detected. Reset is available only to TCP-based attack signatures.
  - ♦ IP Log: The sensor generates an IP session log with information about the attack.

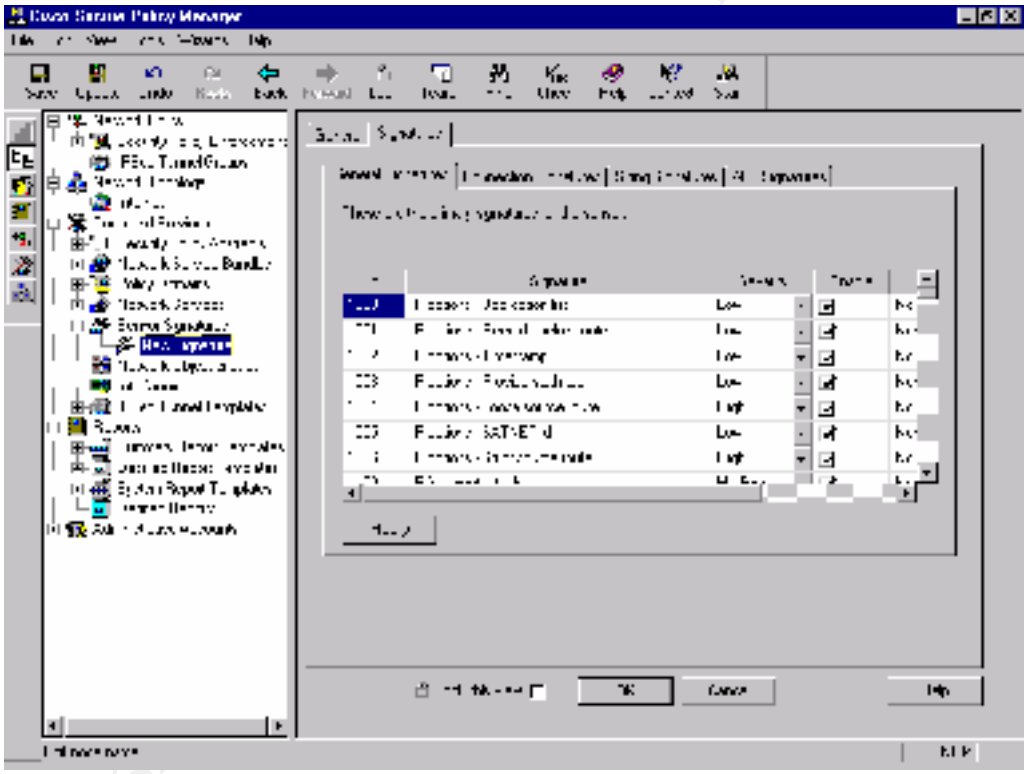

Figure 2 – General Signatures Panel

Configuring Connection Signatures

Connection signatures are user-configurable attack signatures based on the transport-layer protocol (TCP or UDP) and port number of the packets being monitored. You can configure the following properties for each connection signature: (Figure 3)

- Signature: Used to enter a unique name for the attack signature.
- Type: Used to set the protocol layer to TCP or UDP.

- Port: Used to set the port number.
- Severity: Categorizes the attack signature. The severity setting is used in Event Viewer to distinguish among the types of attacks being logged.
- Enable: Configures the sensor to scan network traffic for that particular attack signature and to generate an alarm when the attack is detected. Disabling an attack signature causes the sensor to disregard any network traffic that displays the signature.
- Actions: Determines what action the sensor will take when it detects an attack signature.

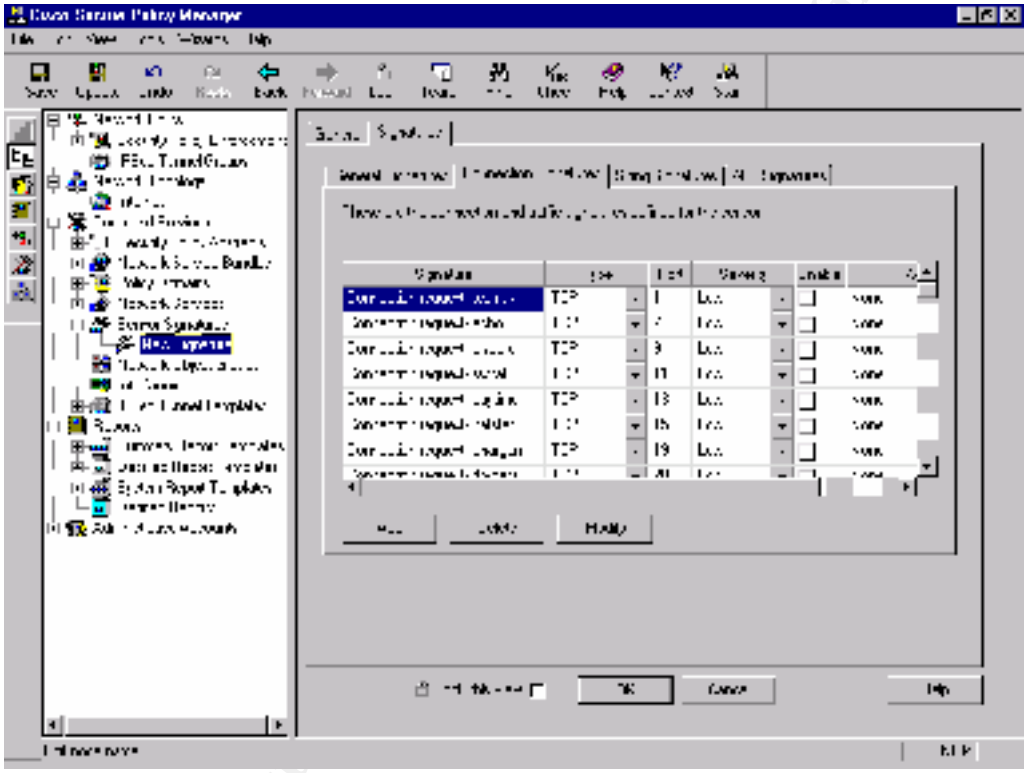

Figure 3 – Connection Signatures Panel

Configuring String-Matching Signatures

String-matching signatures are user-configurable attack signatures based on the content of a particular session. You can configure the following properties for each string-matching signature (Figure 4):

- Signature Characteristics: Defines the string to match, in the form of a regular expression, the port number of the traffic to be monitored, the direction of the traffic, and the number of times a particular string is detected before the sensor generates an alarm.
- Severity: Categorizes the attack signature. The severity setting is used in Event Viewer to distinguish among the types of attacks being logged.
- Enable: Configures the sensor to scan network traffic for that particular attack signature and to generate an alarm when the attack is detected. Disabling an attack signature causes the sensor to disregard any network traffic that displays the signature.
- Actions: Determines what action the sensor will take when it detects an attack signature.

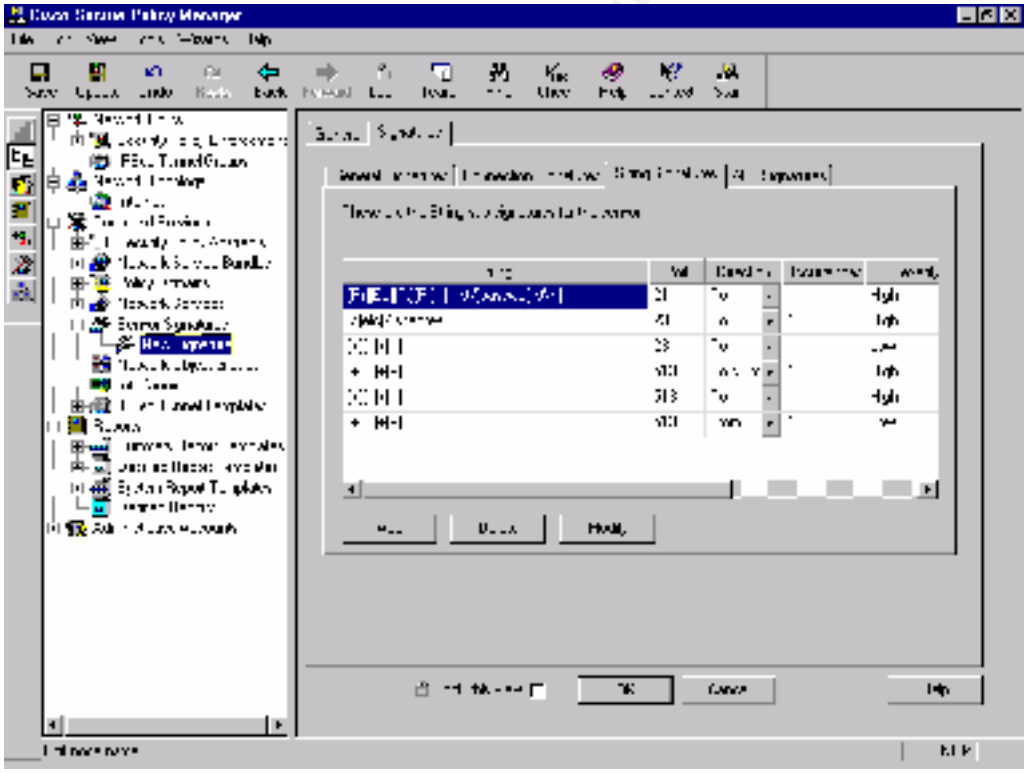

Figure 4 – String-Matching Signatures Panel

Configuring ACL Signatures

ACL signatures are user-configurable attack signatures based on policy violations recorded by network devices in the syslog stream. To configure a sensor to detect ACL signatures, you must first configure one or more routers to log ACL violations. Then, you must configure those routers to communicate with the sensor. Finally, you must configure the sensor to accept syslog traffic from those routers. You can configure the following properties for each ACL signature: (Figure 5)

- Signature Characteristics: The signature characteristics consist of the name or number of the ACL being monitored and a sub-signature ID. You can also provide a comment for the ACL signature.
- Severity: The severity setting categorizes the attack signature. The severity setting is used in Event Viewer to distinguish among the types of attacks being logged.
- Enable: Enabling an attack signature configures the sensor to scan network traffic for that particular attack signature and to generate an alarm when the attack is detected. Disabling an attack signature causes the sensor to disregard any network traffic that displays the signature.
- Actions: Determine what action the sensor will take when it detects an attack signature.

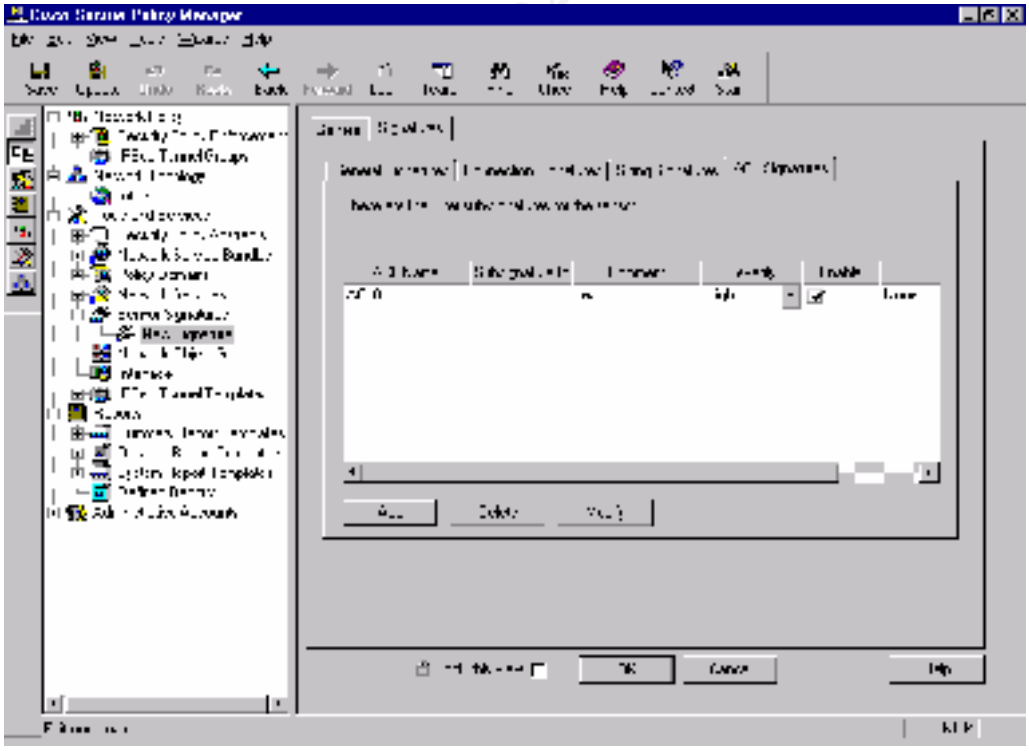

Figure 5 – ACL Signatures Panel

Signature configuration on Sensors

Signature Wizard

From Cisco Secure IDS sensor version 3.0 or later, users can create custom signatures and tune existing signatures directly on the Sensor via the Signature Wizard.

The Signature Wizard allows you to make changes to IDS signatures directly on the Sensor. These same changes can also be made via the version 2.2.3 UNIX Director. CSPM users need the use Signature Wizard on sensors to configure the sensor version 3.0 features.

The Signature Wizard uses the following files in */usr/nr/etc* directory:

- SigData.conf
  Encrypted configuration file that contains all the default signatures. It is updated whenever you install a signature update.
- SigUser.conf
  Configuration file where the engine parameters for signature changes and new signatures are stored. It is automatically updated through the Signature Wizard.
- SigSettings.conf
  Contains global device management (*packetd*) tokens and is automatically updated and managed through the Signature Wizard.

Using the Signature Wizard

Starting Signature Wizard

1. Login to the sensor as user **netrangr**
2. Type **SigWizMenu**, the main menu are shown in Figure 6:

```
Main Menu  :  CSIDS Signature Wizard
------------------------------------------------------------------------------

  Current Sig Data File '/usr/nr/etc/SigData.conf'
  Current Sig User File '/usr/nr/etc/SigUser.conf'
  Current Settings File '/usr/nr/etc/SigSettings.conf'
------------------------------------------------------------------------------

   1 - Tune Signature Parameters
   2 - Add NEW Custom Signature
   3 - Set Custom Signature Severity/Action
   4 - Edit Signature Address Mapping
   5 - Delete Signature Tunings and Custom Signatures
   6 - Other 3.x Tokens
   7 - Display Signatures
   x - EXIT
------------------------------------------------------------------------------

Selection>
```

Figure 6 – Signature Wizard Main Menu

Tuning Existing Signatures

1. Choose "Tune Signature Parameters" from main menu.
2. Enter the Signature ID of the Signature you want to tune.
3. A list of Signature Parameters will be displayed; choose the Parameter that needs to be tuned.
4. Enter a new value for the Signature Parameter, save the change.

Figure 7 shows an example of tuning existing signature (signature ID: 3010 TCP High Port Sweep):

```
Edit Parameters for which Signature ? 3010
Tune Signature Parameters  :  CSIDS Signature Wizard
_____________________________________________________________________

 Current Signature: Engine SWEEP.PORT.TCP SIGID 3010
           SigName: TCP High Port Sweep
_____________________________________________________________________

  0 - Edit ALL Parameters
  1 - AlarmInterval        =
  2 - AlarmThrottle        = FireAll
  3 - ChokeThreshold       = 100
  4 - FlipAddr             = True
  5 - InvertedSweep        = True
  6 - MaxInspectLength     =
  7 - MinHits              =
  8 * PortRange            = 2
  9 - ResetAfterIdle       = 20
 10 - SigComment           =
 11 - SigStringInfo        =
 12 - SupressReverse       = True
 13 - ThrottleInterval     = 30
 14 * Unique               = 25
  d - Delete a value
  u - UNDO and continue
  x - SAVE and continue
_____________________________________________________________________

Selection> 14
Maximum number of unique port connections between the two hosts. (NUMBER : 2
- 40)
- Unique -
[current value] 25
[new value] > 30
```

Figure 7 – Tuning Signature Parameters Menu

In this example, the parameter value of "Maximum number of unique port connections between the two hosts" is changed from 25 to 30.

Creating a New Signature

To add a custom Signature, the following steps have to be done:

- Select an engine
- Type in a signature ID or have it generated automatically
- Specify a signature name

- Insert the signature in the configuration file
- Tune signature parameters of the new custom signature.

Figure 8 shows the Add New Signature Menu.

```
Add New Signature  :  CSIDS Signature Wizard
_____________________________________________________________________

   1 – Engine Name    'Not Set'
   2 – Generate SIGID
   3 – Signature ID   'Not Set'
   4 – Signature Name 'Not Set'
   5 – INSERT NOW
ENTER– BACK TO MAIN
_____________________________________________________________________
selection>
```

Figure 8 – Add New Signature Menu

Creating New Signature Example

We will introduce an example of creating new custom signature.

- Determine what the signature will detect:
  An ICMP custom signature will be used in this example. This signature will detect the ICMP packet with the parameters listed below:
  SEQ=0 TYPE=0 CODE=123
- Determine an engine that covers this kind of attack:
- ATOMIC.ICMP engine will be used. This engine has *IcmpSeq*, *IcmpType* and *IcmpCode* parameters that achieve our requirement.
- Define a signature ID:
  20001. (User-defined signatures are valid in the range: 20,000-50,000)
- Define a name for the signature: ICMP test
- Insert the signature in the configuration file.
- Set alarm level: 3
- Set alarm action: 2 (log)
- Tune new signature parameters: set new values for the parameters:
  IcmpSeq = 0
  IcmpType = 0
  IcmpCode = 123

Figure 9 shows the procedures of creating and tuning new signature:

```
_____________________________________________________________________
Add NEW Custom Signature  :  CSIDS Signature Wizard
_____________________________________________________________________

          1 - Engine Name     'Not Set'
          2 - Generate SIGID
          3 - Signature ID    'Not Set'
          4 - Signature Name  'Not Set'
          5 - INSERT NOW
       ENTER - BACK TO MAIN
_____________________________________________________________________
```

```
Selection> 1
Engine Selection  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

1 - ATOMIC.ICMP       Simple ICMP alarms based on Type, Code, Seq, Id, etc.
2 - ATOMIC.IPOPTIONS  Simple L3 Alarms.
3 - ATOMIC.L3.IP      Simple L3 IP Alarms.
4 - ATOMIC.TCP        Simple TCP packet alarms based on Port, Destination and
                      Flags.
5 - ATOMIC.UDP        Simple UDP packet alarms based on Port, Direction and
                      DataLength.
6 - FLOOD.HOST.ICMP   Icmp Floods directed at a single host
7 - FLOOD.HOST.UDP    Udp Floods directed at a single host
8 - FLOOD.NET         Multi-protocol floods directed at a network segment.
9 - FLOOD.TCPSYN      Connections to multiple ports using TCP SYN.
10 - SERVICE.DNS.TCP  DNS Packet Analyzer on TCP port 53 - includes compres-
                      sion handler
11 - SERVICE.DNS.UDP  UDP based DNS signatures
12 - SERVICE.PORTMAP  RPC Program number sent to Port Mapper.
13 - SERVICE.RPC      Simple RPC alarms based on Program, Procedure, Length.
14 - STRING.HTTP      Specialized STRING.TCP alarms for web traffic. Includes
                      anti-evasive URL deobfuscation.
15 - STRING.ICMP      Generic ICMP based string search Engine.
16 - STRING.TCP       Generic TCP based string search Engine.
17 - STRING.UDP       Generic UDP based string search Engine.
18 - SWEEP.HOST.ICMP  A single host sweeping a range of nodes using ICMP.
19 - SWEEP.HOST.TCP   A single host sweeping a range of nodes using TCP.
20 - SWEEP.PORT.TCP   TCP connections to multiple destination ports between
                      two nodes.
21 - SWEEP.PORT.UDP   UDP connections to multiple destination ports between
                      two nodes.
22 - SWEEP.RPC        Connections to multiple ports with RPC requests between
                      two nodes.
ENTER - Back
-------------------------------------------------------------------------------

Selection> 1
Add NEW Custom Signature  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

           1 - Engine Name     'ATOMIC.ICMP'
           2 - Generate SIGID
           3 - Signature ID    'Not Set'
           4 - Signature Name  ''
           5 - INSERT NOW
       ENTER - BACK TO MAIN
-------------------------------------------------------------------------------

Selection> 3
ENTER Signature ID Number [between 20000-50000]
20001
Add NEW Custom Signature  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

           1 - Engine Name     'ATOMIC.ICMP'
           2 - Generate SIGID
           3 - Signature ID    '20001'
           4 - Signature Name  ''
           5 - INSERT NOW
       ENTER - BACK TO MAIN
-------------------------------------------------------------------------------

Selection> 4
```

```
ENTER Signature Name > ICMP test
Add NEW Custom Signature  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

          1 - Engine Name     'ATOMIC.ICMP'
          2 - Generate SIGID
          3 - Signature ID    '20001'
          4 - Signature Name  'ICMP test'
          5 - INSERT NOW
      ENTER - BACK TO MAIN
-------------------------------------------------------------------------------

Selection> 5
Adjust Severity and Action  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

Signature:    20001
Alarm Level:  0  (OFF)
Alarm Action: 0  None
-------------------------------------------------------------------------------

     0 - Turn signature OFF
     1 - set Alarm Severity 1
     2 - set Alarm Severity 2
     3 - set Alarm Severity 3
     4 - set Alarm Severity 4
     5 - set Alarm Severity 5
 ENTER - adjust Action
     x - DONE
-------------------------------------------------------------------------------

Selection> 3
Adjust Severity and Action  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

Signature:    20001
Alarm Level:  3
Alarm Action: 0  None
-------------------------------------------------------------------------------

     0 - set Action NONE
     1 - set Action Shun
     2 - set Action Log
     3 - set Action Shun & Log
     4 - set Action Reset
     5 - set Action Shun & Reset
     6 - set Action Log & Reset
     7 - set Action Shun & Log & Reset
 ENTER - adjust Severity
     x - DONE
-------------------------------------------------------------------------------

Selection> 2
Adjust Severity and Action  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

Signature:    20001
Alarm Level:  3
Alarm Action: 2  Log
-------------------------------------------------------------------------------

     0 - Turn signature OFF
     1 - set Alarm Severity 1
     2 - set Alarm Severity 2
```

```
     3 - set Alarm Severity 3
     4 - set Alarm Severity 4
     5 - set Alarm Severity 5
 ENTER - adjust Action
     x - DONE
-------------------------------------------------------------------------------

Selection> x
Tune Signature Parameters  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

 Current Signature: Engine ATOMIC.ICMP SIGID 20001
           SigName: ICMP test
-------------------------------------------------------------------------------

  0 - Edit ALL Parameters
  1 - AlarmThrottle        = Summarize
  2 - ChokeThreshold       = 100
  3 - FlipAddr             =
  4 - IcmpCode             =
  5 - IcmpId               =
  6 - IcmpMaxCode          =
  7 - IcmpMaxSeq           =
  8 - IcmpMinCode          =
  9 - IcmpMinSeq           =
 10 - IcmpSeq              =
 11 - IcmpType             =
 12 - IpTOS                =
 13 - MaxInspectLength     =
 14 - MinHits              =
 15 - ResetAfterIdle       = 15
 16 - SigComment           =
 17 - SigName              = ICMP test
 18 - SigStringInfo        =
 19 - ThrottleInterval     = 30
 20 - WantFrag             =
  d - Delete a value
  u - UNDO and continue
  x - SAVE and continue
-------------------------------------------------------------------------------

Selection> 10

 ICMP header SEQUENCE value (NUMBER)

IcmpSeq =   > 0

Tune Signature Parameters  :  CSIDS Signature Wizard
-------------------------------------------------------------------------------

 Current Signature: Engine ATOMIC.ICMP SIGID 20001
           SigName: ICMP test
-------------------------------------------------------------------------------

  0 - Edit ALL Parameters
  1 - AlarmThrottle        = Summarize
  2 - ChokeThreshold       = 100
  3 - FlipAddr             =
  4 - IcmpCode             =
  5 - IcmpId               =
  6 - IcmpMaxCode          =
  7 - IcmpMaxSeq           =
  8 - IcmpMinCode          =
  9 - IcmpMinSeq           =
 10 - IcmpSeq              = 0
```

```
 11 - IcmpType             =
 12 - IpTOS                =
 13 - MaxInspectLength     =
 14 - MinHits              =
 15 - ResetAfterIdle       = 15
 16 - SigComment           =
 17 - SigName              = ICMP test
 18 - SigStringInfo        =
 19 - ThrottleInterval     = 30
 20 - WantFrag             =
  d - Delete a value
  u - UNDO and continue
  x - SAVE and continue
--------------------------------------------------------------------------------

Selection> 11

 ICMP header TYPE value (NUMBER)

IcmpType =   > 0

Tune Signature Parameters  :  CSIDS Signature Wizard
--------------------------------------------------------------------------------

 Current Signature: Engine ATOMIC.ICMP SIGID 20001
           SigName: ICMP test
--------------------------------------------------------------------------------

  0 - Edit ALL Parameters
  1 - AlarmThrottle        = Summarize
  2 - ChokeThreshold       = 100
  3 - FlipAddr             =
  4 - IcmpCode             =
  5 - IcmpId               =
  6 - IcmpMaxCode          =
  7 - IcmpMaxSeq           =
  8 - IcmpMinCode          =
  9 - IcmpMinSeq           =
 10 - IcmpSeq              = 0
 11 - IcmpType             = 0
 12 - IpTOS                =
 13 - MaxInspectLength     =
 14 - MinHits              =
 15 - ResetAfterIdle       = 15
 16 - SigComment           =
 17 - SigName              = ICMP test
 18 - SigStringInfo        =
 19 - ThrottleInterval     = 30
 20 - WantFrag             =
  d - Delete a value
  u - UNDO and continue
  x - SAVE and continue
--------------------------------------------------------------------------------

Selection> 4

 ICMP header CODE value (NUMBER)

IcmpCode =   > 123

Tune Signature Parameters  :  CSIDS Signature Wizard
--------------------------------------------------------------------------------

 Current Signature: Engine ATOMIC.ICMP SIGID 20001
           SigName: ICMP test
```

```
  0 - Edit ALL Parameters
  1 - AlarmThrottle        = Summarize
  2 - ChokeThreshold       = 100
  3 - FlipAddr             =
  4 - IcmpCode             = 123
  5 - IcmpId               =
  6 - IcmpMaxCode          =
  7 - IcmpMaxSeq           =
  8 - IcmpMinCode          =
  9 - IcmpMinSeq           =
 10 - IcmpSeq              = 0
 11 - IcmpType             = 0
 12 - IpTOS                =
 13 - MaxInspectLength     =
 14 - MinHits              =
 15 - ResetAfterIdle       = 15
 16 - SigComment           =
 17 - SigName              = ICMP test
 18 - SigStringInfo        =
 19 - ThrottleInterval     = 30
 20 - WantFrag             =
  d - Delete a value
  u - UNDO and continue
  x - SAVE and continue

Selection> x
```

Figure 9 – Creating and Tuning New Signature Procedures

Signature Tuning Recommendation

In general, network malicious activities can be divided into three categories:

- Reconnaissance activity
- Denial of service
- Specific service attack

Most of reconnaissance activities are focused on information gathering of your network. Although these activities might be preludes of more serious attacks, there is no immediate destructive impact on your network. Furthermore, reconnaissance traffic is very difficult to analyze, and has a very low percentage of true attacks. Therefore, it is recommended to disabled some of reconnaissance signatures.

Signatures recommended to be disabled

Signatures listed below could be triggered by normal network traffic or reconnaissance network activities. To reduce false positives, we suggest disabling the following signatures:

- 2000 ICMP Echo Reply
- 2001 ICMP Host Unreachable
- 2004 ICMP Echo Request
- 2005 ICMP Time Exceeded for a Datagram

- 3301 NETBIOS Stat
- 3302 NETBIOS Session Setup Failure
- 3304 Windows Null Account Name
- 3309 Windows SRVSVC Access
- 4002 UDP Flood
- 6901 Net Flood ICMP Reply
- 6902 Net Flood ICMP Request

Summary

To achieve the best performance, Intrusion Detection Systems need to be tuned based on different network traffic and IDS implementation. Signatures tuning plays an important role in IDS tuning.

Currently, there are about 400 Cisco Secure IDS signatures. “Signature Wizard” on the Cisco Secure IDS sensor provides flexibility and convenience for users to perform Signatures tuning and creation. We consider “Signature Wizard” as a powerful tool of signatures customization and look forward to more improvements in the future.

Reference

Cisco Secure Policy Manager Solution Guide Series: Intrusion Detection System, Version 2.3.1i.
http://www.cisco.com/univercd/cc/td/doc/product/ismg/policy/ver23i/idsguide/index.htm

Cisco Intrusion Detection System Signature Engines Version 3.0
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids6/13346_01.htm

Cisco Intrusion Detection System Sensor Configuration Note Version 3.0
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids6/12216_02.htm

Cisco Secure Intrusion Detection System Internal Architecture
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/0866_02.htm

CSPM Reference Guide
http://www.cisco.com/warp/public/cc/pd/sqsw/sqppmn/prodlit/csp22_rg.htm

Assignment 2 – Network Detects

Detect 1 – Nimda Worm

```
PART 1

Nov 12 02:26:12 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1181 -> a.b.c.62:80
Nov 12 02:26:12 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:3828 -> a.b.c.62:80
Nov 12 02:26:15 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1198 -> a.b.c.62:80
Nov 12 02:26:18 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1425 -> a.b.c.62:80
Nov 12 02:26:22 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:2133 -> a.b.c.62:80
Nov 12 02:26:25 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:2864 -> a.b.c.62:80
Nov 12 02:26:35 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1559 -> a.b.c.62:80
Nov 12 02:26:35 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:3125 -> a.b.c.62:80
Nov 12 02:26:36 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1573 -> a.b.c.62:80
Nov 12 02:26:45 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:1639 -> a.b.c.62:80
Nov 12 02:26:46 hosthu snort: [1:1113:1] WEB-MISC http directory traversal
[Classification: Attempted Information Leak] [Priority: 2]: {TCP}
24.169.176.197:3250 -> a.b.c.62:80

PART 2

24.169.176.197 - - [12/Nov/2001:02:25:58 -0500] "GET /scripts/root.exe?/c+dir
HTTP/1.0" 404 289 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:01 -0500] "GET /MSADC/root.exe?/c+dir
HTTP/1.0" 404 287 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:05 -0500] "GET
/c/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 297 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:08 -0500] "GET
/d/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 297 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:11 -0500] "GET
/scripts/..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 311 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:12 -0500] "GET
/_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0" 404 328 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:15 -0500] "GET
/_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0" 404 328 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:18 -0500] "GET
/msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%c1%1c../winnt/sys
tem32/cmd.exe?/c+dir HTTP/1.0" 404 344 "-" "-"
```

```
24.169.176.197 - - [12/Nov/2001:02:26:22 -0500] "GET
/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:25 -0500] "GET
/scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:35 -0500] "GET
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:35 -0500] "GET
/scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:35 -0500] "GET
/scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 294 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:45 -0500] "GET
/scripts/..%%35c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 294 "-" "-"
24.169.176.197 - - [12/Nov/2001:02:26:46 -0500] "GET
/scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 311 "-" "-
"
```

1. Source of Trace:

The traces were collected from incidents.org:
http://www.incidents.org/archives/intrusions/msg02431.html

2. Detect was generated by:

The first part of this detect was generated by Snort IDS, which sent the alert messages to syslog in the following format:

[Date & Time] [Hostname] Snort: [Generator ID: Snort ID: Revision number] [Message] [Priority] {Protocol} [Source IP: port -> Destination IP: port]

The second part of this detect was from Apache web server access log, with the following format:

[Client IP] - - [Time & Date] [HTTP request] [Server status code] [Object size returned to the client]

This detect was generated from 12/Nov/2001:02:25:58 to 12/Nov/2001:02:26:46 (i.e. 48 seconds).

3. Probability the source address was spoofed:

Low. The attacker must complete TCP three-way handshake with the web server before sending http request to it. Furthermore, the attacker attempted to execute system commands on the web server and had to rely on the responses from the web server.

4. Description of attack:

The attacker tried to gain system control by exploiting Microsoft IIS vulnerabilities on the web server, including “IIS/PWS Extended Unicode Directory Transversal Vulnerability” and “IIS/PWS Escape Character Decoding Command Execution Vulnerability”. It also tried to exploit the root.exe backdoor left by Code Red II or Sadmind infections.

5. Attack mechanism:

Nimda worm compromises the security of infected hosts. It provides remote attackers full access and control over the infected hosts. Nimda propagates itself through four distinct mechanisms:

- It scans Internet for web servers and attempts to exploit Microsoft web server vulnerabilities to get full control of victim hosts.
- It forwards itself to other email addresses found on the victim host. It uses "Microsoft IE MIME Header Attachment Execution Vulnerability" to send out HTML emails
- On infected web servers, Nimda uses http service to propagate itself to the clients that browse the infected web server. Again, it uses the exploitation that IE handles unusual MIME type incorrectly.
- The virus copies itself into the Windows directory with the filenames load.exe and riched20.dll, and attempts to spread itself to other users via network shares.

This detect shows the first propagating mechanism. The analysis for the http requests are shown as following:

```
GET /scripts/root.exe?/c+dir HTTP/1.0
GET /MSADC/root.exe?/c+dir HTTP/1.0
```

Nimda attempts to exploit the root.exe backdoor left by Code Red II and Sadmind infections.

```
GET /c/winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /d/winnt/system32/cmd.exe?/c+dir HTTP/1.0
```

These two attacks are targeting at Code Red II backdoors where root C: and D: drives are mapped to IIS virtual folders.

```
GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir HTTP/1.0
```

These http requests are used to exploit the "IIS/PWS Extended Unicode Directory Transversal Vulnerability". With this vulnerability, Remote users can execute arbitrary commands with the privileges of the IUSR_*machinename* account.

```
GET /scripts/..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET /_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir
HTTP/1.0
GET
/msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%c1%1c../winnt/sys
tem32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%%35c../winnt/system32/cmd.exe?/c+dir HTTP/1.0
GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0
```

These http requests are used to exploit "IIS/PWS Escape Character Decoding Command Execution Vulnerability". Because the web server will decode the requested pathname twice, Nimda uses double encoding to pass the security check after the first decoding.

6. Correlations:

Similar Nimda worm detects can be found on incidents.org website: http://www.incidents.org/archives/intrusions/msg02599.html

IIS/PWS Extended Unicode Directory Transversal Vulnerability:

Bugtraq ID: 1806 / CVE ID: CVE-2000-0884
Microsoft Security Bulletin MS00-078
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-078.asp
VulDB: http://www.securityfocus.com/bid/1806

IIS/PWS Escape Character Decoding Command Execution Vulnerability:

Bugtraq ID: 2708 / CVE ID: CAN-2001-0333
Microsoft Security Bulletin MS01-026
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-026.asp
VulDB: http://www.securityfocus.com/bid/2708

Microsoft IE MIME Header Attachment Execution Vulnerability:

Bugtraq ID: 2524 / CVE ID: CAN-2001-0154
Microsoft Security Bulletin MS01-020
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-020.asp
VulDB: http://www.securityfocus.com/bid/2524

7. Evidence of active targeting:

Nimda worm first scans Internet for web servers, then targets the web servers. In this detect, the web server was targeted.

8. Severity

Component	Score	Comment
Target Criticality	4	Apache Web server
Attack Lethality	4	Nimda worm can get full control of the victim host
System Countermeasures	4	Apache web server is not vulnerable to IIS vulnerabilities
Network Countermeasures	3	Snort IDS
Attack Severity	1	

Severity score = (Criticality + Lethality) – (System + Net Countermeasures)

9. Defensive recommendation

The defensive setup is fine; however, it is recommended that the system should be reviewed periodically to ensure it is running the latest version/patch/rule of the application/operating system.

10. Multiple choice test question:

Which of the following is NOT a propagation mechanism of Nimda worm?

a) Forward itself as an attachment in the HTML email

b) Unicode directory transversal vulnerability

c) FTP

d) Copy it through network shares

Answer: C. Nimda propagates through four distinct mechanisms; please refer to “Attack Mechanism” for the details.

Detect 2 – Port Scan on TCP 1024 and 3072

```
10/30-04:37:34.362796 172.157.50.40:0 -> a.b.158.66:3072
TCP TTL:110 TOS:0x0 ID:57953 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x978D7733  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:45.864666 172.157.50.40:0 -> a.b.122.50:3072
TCP TTL:42 TOS:0x0 ID:36537 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x23CBE351  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:47.152899 172.157.50.40:0 -> a.b.209.50:3072
TCP TTL:42 TOS:0x0 ID:13960 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x7B0D0E2C  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:47.903162 172.157.50.40:0 -> a.b.213.58:3072
TCP TTL:42 TOS:0x0 ID:39113 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0xCF1B1A  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:50.271866 172.157.50.40:0 -> a.b.87.100:3072
TCP TTL:42 TOS:0x0 ID:55770 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0xA734827F  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:51.191331 172.157.50.40:0 -> a.b.12.58:3072
TCP TTL:42 TOS:0x0 ID:21835 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0xD85E986A  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:51.768250 172.157.50.40:0 -> a.b.182.14:1024
TCP TTL:42 TOS:0x0 ID:41898 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0xB1409D80  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:53.004036 172.157.50.40:0 -> a.b.36.71:3072
TCP TTL:42 TOS:0x0 ID:18856 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x521B2B57  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:54.066091 172.157.50.40:0 -> a.b.128.16:3072
TCP TTL:42 TOS:0x0 ID:55620 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x2BE5C64E  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:55.405133 172.157.50.40:0 -> a.b.70.93:1024
TCP TTL:42 TOS:0x0 ID:37190 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0xFF3BE604  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:56.672060 172.157.50.40:0 -> a.b.119.64:1024
TCP TTL:42 TOS:0x0 ID:14284 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x254C7E3E  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:58.004546 172.157.50.40:0 -> a.b.127.19:1024
TCP TTL:42 TOS:0x0 ID:60359 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x881F300C  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:38:58.093051 172.157.50.40:0 -> a.b.221.12:1024
TCP TTL:42 TOS:0x0 ID:63338 IpLen:20 DgmLen:40
```

```
***A*R** Seq: 0x0  Ack: 0x6614443A  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

10/30-04:39:12.043996 172.157.50.40:0 -> a.b.222.27:1024
TCP TTL:42 TOS:0x0 ID:42154 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x60A7FD61  Win: 0x0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

1. Source of Trace:

The traces were collected from incidents.org:
http://www.incidents.org/archives/intrusions/msg02326.html

2. Detect was generated by:

This detect was generated by Snort IDS with the following format:

[Date & time][Source address & port]->[Destination address & port] [Protocol][Time to Live][Type of service][IP header length] [Datagram length][TCP flags set] [Sequence number][Acknowledge number][Window size][TCP header length]

This detect was generated from Oct 30 04:37:34 to Oct 30 04:39:12.

3. Probability the source address was spoofed:

Likely. Normal traffic should not have source or destination port set to zero. It is also possible a third party effect – some people tried to connect to inexistent port 0 on the host 172.157.50.40, and the host simply replied them with a TCP reset.

The host 172.157.50.40 (AC9D3228.ipt.aol.com) appears to be an America online dynamic IP address, which is very difficult to trace back.

4. Description of attack:

The attacker sent TCP reset packets to multiple hosts targeted on port 1024 and 3072. It looks like the attacker was probing port 1024 (Trojan Jade, Latinus, NetSpy) and 3072 (ContinuStor Monitor Port) on multiple hosts. However, it could also be the reply packets to inexistent service port – TCP port 0.

5. Attack mechanism:

There are two possible attack mechanisms for this detect:

1. A distributed denial of service attack targeted the host 172.157.50.40 on TCP port 0, and the host just replied them with TCP reset packets, since it doesn't listen on that port at all.
2. The attacker crafted the TCP packets with source port 0 and RST/ACK flags and tried to bypass dummy packet filters. If it had succeeded, it would start another probe or attack toward the victim hosts.

6. Correlations:

Similar detect can be found on the following website:

http://lists.insecure.org/firewall-wizards/2001/May/0052.html

http://www.sans.org/y2k/120400-1100.htm

7. Evidence of active targeting:

If this detect was distributed denial of service against the host 172.157.50.40, it was definitely an active targeting.

If it was attack mechanism #2, the attacker probed multiple hosts on TCP port 1024 and 3072; then it did not appear to be an attack targeting at any host.

8. Severity

Attack mechanism #1

Component	Score	Comment
Target Criticality	2	Specific host
Attack Lethality	4	DDoS attack
System Countermeasures	0	Need more information
Network Countermeasures	0	Need more information
Attack Severity	5	

Attack mechanism #2

Component	Score	Comment
Target Criticality	1	Random hosts
Attack Lethality	2	Port scan
System Countermeasures	0	Need more information
Network Countermeasures	0	Need more information
Attack Severity	3	

Severity score = (Criticality + Lethality) – (System + Net Countermeasures)

9. Defensive recommendation

Based on the logs, we don't have enough information about the system and network countermeasure. But for the distributed denial of service attacks, it is strongly recommended to install a perimeter devise (access router or firewall) that can apply basic access control and filter out unrelated traffic.

10. Multiple choice test question:

```
10/30-04:39:12.043996 172.157.50.40:0 -> a.b.222.27:1024
TCP TTL:42 TOS:0x0 ID:42154 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x60A7FD61  Win: 0x0  TcpLen: 20
```

What is abnormal in this Snort log?

a) TCP flags: RST/ACK

b) Sequence number 0

c) Source port 0

d) Window size 0

Answer: C, normal traffic does not have source or destination port set to 0.

Detect 3 – FTP Probes

```
Nov 12 21:54:04 host1 proftpd[29464] host1 (y1-1a247.neo.rr.com
[24.93.246.247]): FTP session opened.
```

```
Nov 12 21:54:05 hostl proftpd[29464] hostl (y1-1a247.neo.rr.com
[24.93.246.247]): ANON anonymous: Login successful.
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "CWD /incoming/"
250 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "CWD /pub/" 250
-
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "CWD
/pub/incoming/" 550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "CWD /public/"
550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "MKD
011112214953p" 550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "MKD
011112214953p" 550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "PASS
guest@here.com" 230 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:09 -0500] "CWD /" 250 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:09 -0500] "CWD /_vti_pvt/"
550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:10 -0500] "CWD /upload/"
550 -
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:10 -0500] "MKD
011112214957p" 550 -
Nov 12 21:54:10 hostl proftpd[29464] hostl (y1-
1a247.neo.rr.com[24.93.246.247]): FTP session closed.

Nov 12 21:54:52 hostca in.ftpd[19941]: refused connect from y1-
1a247.neo.rr.com
Nov 12 21:54:52 hostca in.ftpd[19942]: refused connect from y1-
1a247.neo.rr.com
Nov 12 21:54:52 hostca in.ftpd[19943]: refused connect from y1-
1a247.neo.rr.com
Nov 12 21:54:04 hostt ftpd[2429]: refused connect from y1-1a247.neo.rr.com
Nov 12 21:54:52 hostca in.ftpd[19944]: refused connect from y1-
1a247.neo.rr.com
Nov 12 21:54:05 hostsa ftpd[11122]: refused connect from y1-1a247.neo.rr.com
Nov 12 21:54:05 hostz ftpd[544]: refused connect from y1-1a247.neo.rr.com
Nov 12 21:57:56 hostmau Connection attempt to TCP z.y.w.12:21 from
24.93.246.247:2145
Nov 12 21:57:57 hostmau Connection attempt to TCP z.y.w.12:21 from
24.93.246.247:2145
Nov 12 21:57:58 hostmau Connection attempt to TCP z.y.w.12:21 from
24.93.246.247:2145
```

1. Source of Trace:

The traces were collected from incidents.org:
http://www.incidents.org/archives/intrusions/msg02432.html

2. Detect was generated by:

This detect was the combination of Unix message log and Proftpd log with the following format:

Proftpd Log Format:

[Remote hostname][Remote username][Local authentication userid][Current local time][Full command line received from client][Numeric FTP response code][Bytes sent for request]

Unix Message Log Format:

[Date & time][Hostname][Daemon/program][Process id][Messages]

This detect were collected from Nov 12 21:54:04 to Nov 12 21:57:58.

3. Probability the source address was spoofed:

Low. The attacker tried to login to the ftp server using "ftp" or "anonymous" user ID, which relied on the responses from ftp server. Once it logged in successfully, it executed "CWD" and "MKD" commands, which is very difficult to achieve using spoofing IP address.

4. Description of attack:

The attacker scanned ftp servers on the Internet, then tried to login to the ftp server with "anonymous" or "ftp" user ID. Once the attacker successfully logged in, it executed a series of commands to find specific directories and tried to make new directory "011112214953p" and "011112214957p".

5. Attack mechanism:

Based on the Proftpd logs, the attacker executed several CWD and MKD commands in a second, which obviously cannot be done manually; therefore, it is very likely this ftp scans was generated by automated tools such as Synscan and Sscan2k-pre6.

The commands executed on the ftp server are analyzed as following:

```
CWD /incoming/
CWD /pub/
CWD /pub/incoming/
CWD /public/
CWD /
CWD /upload/
```

These commands can be explained that the attacker attempted to find the upload directories that can be used for another attack, since in most cases these directories are open to public read/write access.

```
CWD /_vti_pvt/
```

It is possible the attacker was looking for the Microsoft FrontPage configuration files (password file) in the /vti_pvt directory. A successful attempt of this probe also indicates that the web server root is accessible via anonymous FTP, which is very insecure to the ftp server.

```
MKD 011112214953p
MKD 011112214957p
```

The MKD commands indicate the attacker attempted to create directory to store large files. Also, if the attacker is able to create directories, it may be possible to exploit a wu-ftpd vulnerability to gain root access through a series of MKD and CWD commands (CVE-1999-0950).

The FTP server was not vulnerable to these exploits in this detect.

6. Correlations:

Similar detects can be found in the following websites:

http://www.incidents.org/archives/intrusions/msg00952.html

http://www.incidents.org/archives/intrusions/msg02598.html

http://www.incidents.org/archives/intrusions/msg01981.html

These correlation logs consist of Unix message logs, Proftpd logs and Snort IDS portscan logs.

FTP Server remotely exploitable buffer overflow vulnerability (CVE-1999-0950):

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0950

7. Evidence of active targeting:

In this detect, the attacker scanned many ftp servers for anonymous logins, and this detect was generated by automated tools, therefore, it did not appear to be an attack targeted at any host.

8. Severity

Component	Score	Comment
Target Criticality	3	FTP server
Attack Lethality	4	FTP scans and wu-ftpd vulnerability
System Countermeasures	3	FTP server accepts anonymous login but not vulnerable
Network Countermeasures	0	Need more information
Attack Severity	4	

Severity score = (Criticality + Lethality) – (System + Net Countermeasures)

9. Defensive recommendation

It is recommended to disable ftp anonymous login. And always remember to apply the latest patch on the ftp server.

10. Multiple choice test question:

```
y1-1a247.neo.rr.com UNKNOWN ftp [12/Nov/2001:21:54:05 -0500] "CWD /public/"
550 -
```

In this Proftpd log, what does “550” mean?

a) Command not implemented

b) Requested action not taken. File unavailable.

c) Command okay

d) Can't open data connection

Answer: B. The number is FTP reply (status) code.

Detect 4 – SYN Scan from/to TCP port 22

```
PART 1
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.71:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.14:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.62:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.182:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.195:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.222:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.183:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.72:22 SYN ******S*
```

```
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.76:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.190:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.245:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.73:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.191:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.198:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.199:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.201:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.203:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.235:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.247:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.249:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.d.250:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.43:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.72:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.79:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.116:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.179:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.184:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.207:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.229:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.247:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.29:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.208:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.210:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.219:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.e.228:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.14:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.16:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.20:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.32:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.49:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.153:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.178:22 SYN ******S*
Nov 12 16:44:45 195.14.253.217:22 -> a.b.f.190:22 SYN ******S*
Nov 12 16:44:49 195.14.253.217:2408 -> a.b.c.62:22 SYN ******S*

PART 2
Nov 12 16:44:49 hosthu sshd[751]: Did not receive ident string from
195.14.253.217.
Nov 12 16:44:49 hosthu sshd[752]: Did not receive ident string from
195.14.253.217.
Nov 12 16:47:28 hostdar sshd1[17754]: refused connect from 195.14.253.217
Nov 12 16:47:28 hostdar sshd1[17755]: refused connect from 195.14.253.217
Nov 12 16:47:29 hostdar sshd1[17756]: refused connect from 195.14.253.217
Nov 12 16:47:29 hostdar sshd1[17757]: refused connect from 195.14.253.217

Nov 13 01:18:46 hostbe sshd1[1552]: refused connect from 195.14.253.217
Nov 13 01:18:46 hostbe sshd1[1553]: refused connect from 195.14.253.217
Nov 13 01:18:46 hostbe sshd1[1554]: refused connect from 195.14.253.217
Nov 13 01:18:49 hostt sshd[13725]: refused connect from 195.14.253.217
Nov 13 01:18:50 hostcr sshd[21606]: Did not receive identification string
from 195.14.253.217.
Nov 13 01:18:50 hostro sshd1[29452]: connect from 195.14.253.217
Nov 13 01:18:50 hostsa sshd[9382]: refused connect from 195.14.253.217
Nov 13 01:19:14 hostj sshd1[9171]: refused connect from 195.14.253.217
Nov 13 01:19:14 hostj sshd1[9172]: refused connect from 195.14.253.217
Nov 13 01:19:14 hostj sshd1[9173]: refused connect from 195.14.253.217
Nov 13 01:19:14 hostmi sshd[5821]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
Nov 13 01:19:14 hostmi sshd[5822]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
Nov 13 01:19:14 hosty sshd[25150]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
```

```
Nov 13 01:19:14 hosty sshd[25151]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
Nov 13 01:19:15 hostmi sshd[5823]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
Nov 13 01:19:15 hosty sshd[25152]: [ID 800047 auth.info] Did not receive
identification string from 195.14.253.217.
Nov 13 01:19:23 hoste sshd[35131]: twist 195.14.253.217 to /bin/echo "You are
not welcome to use sshd from 195.14.253.217."
Nov 13 01:19:37 hostca sshd[20078]: Denied connection from 195.14.253.217 by
tcp wrappers.
Nov 13 01:19:37 hostca sshd[20078]: WARNING: Denied connection from
195.14.253.217 by tcp wrappers.
Nov 13 01:19:37 hostca sshd[20078]: refused connect from 195.14.253.217
Nov 13 01:19:37 hostca sshd[20079]: Denied connection from 195.14.253.217 by
tcp wrappers.
Nov 13 01:19:37 hostca sshd[20079]: WARNING: Denied connection from
195.14.253.217 by tcp wrappers.
Nov 13 01:19:37 hostca sshd[20079]: refused connect from 195.14.253.217
Nov 13 01:19:37 hostca sshd[20080]: Denied connection from 195.14.253.217 by
tcp wrappers.
Nov 13 01:19:37 hostca sshd[20080]: WARNING: Denied connection from
195.14.253.217 by tcp wrappers.
Nov 13 01:19:37 hostca sshd[20080]: refused connect from 195.14.253.217
Nov 13 01:19:38 hostca sshd[20081]: Denied connection from 195.14.253.217 by
tcp wrappers.
Nov 13 01:19:38 hostca sshd[20081]: WARNING: Denied connection from
195.14.253.217 by tcp wrappers.
Nov 13 01:19:38 hostca sshd[20081]: refused connect from 195.14.253.217
Nov 13 01:22:44 hostmau sshd[12955]: refused connect from 195.14.253.217
(195.14.253.217)
```

1. Source of Trace:

The traces were collected from incidents.org:
http://www.incidents.org/archives/intrusions/msg02432.html

2. Detect was generated by:

The first part of this detect was generated by Snort IDS portscan logs with the following format:

[Date & time] [Source IP: port -> destination IP: port] [TCP Flags info/protocol] [TCP flags set]

The second part of this detect was Unix message log with the following format:

[Date & time][Hostname][Daemon/program][Process id][Messages]

This detect was collected on November 12 and 13.

3. Probability the source address was spoofed:

Low. The attacker performed the SYN scans to multiple hosts on TCP port 22; Once a SSH server was identified, the attacker tried to initiate a real SSH connection to the server. Therefore, probability the source address was spoofed is low.

4. Description of attack:

This detect appears to be a reconnaissance effort against SSH servers on the Internet. Because of the source and destination ports are identical in the

connection, it's very possible this detect was generated by the automated scanning tool.

According to Donald Smith's GCIA practical assignment, scanning tools such as Synscan, T0rnscan and Sscan2k-pre6 would generate packets with identical source and destination ports during port scanning, and use source ports greater than 1024 during connection for vulnerability scan to well-known ports. We believe this detect was generated by one of these tools.

5. Attack mechanism:

When attacker starts a SYN scan, the responses could be divided into the following conditions:

- Destination hosts listens on request port
  If the destination host offers the requested service, it will reply it with a SYN/ACK packet. The attacker then can gather the available services and sequence number information of the destination host. The scanning tool looks for SSH servers by running port scanning on the Internet, and it can perform vulnerability scanning later on these specific hosts.
- Destination hosts not listening on request port
  If the destination host does not provide the requested service, it will reply it with a RST/ACK packet. The attacker knows the service is not available.
- Destination hosts doesn't exist
  When the destination host does not exist, the intermediate router will send an "ICMP: host xyz unreachable" message back to the attacker. The attacker knows the host does not exist.
- Destination port blocked
  If the destination port is blocked by intermediate router, the router will send back an "ICMP: xyz unreachable – admin prohibited filter" message to the attacker. The attacker then knows the port is blocked.
- Destination port blocked, router doesn't respond
  The attacker will keep sending retries until it exhausted the maximum number of retries. This would be the most secure scenario, because the attacker could not get any information from the destination host.

```
Part1
(skip)
Nov 12 16:44:45 195.14.253.217:22 -> a.b.c.62:22 SYN ******S*
(skip)
Nov 12 16:44:49 195.14.253.217:2408 -> a.b.c.62:22 SYN ******S* //last scan

Part 2
(skip)
```

Notice the last scan in the Part 1 detect shows the attacker sent a SYN packet to host a.b.c.62 using source port 2408. This should be the vulnerability scan on the specific host a.b.c.62 after the port scanning.

6. Correlations:

This detect indicates between November 12 and 13, many SSH servers had received the probes from the same attacker 195.14.253.217.

Furthermore, similar detects can be found in the following website:

http://www.incidents.org/archives/intrusions/msg02612.html

7. Evidence of active targeting:

The attacker used the automated tool to perform SYN scans on multiple hosts on the Internet. It did not appear to be an attack targeted on specific host.

8. Severity

Component	Score	Comment
Target Criticality	3	SSH servers
Attack Lethality	3	Reconnaissance attack, real attempt
System Countermeasures	4	TCP Wrappers, SSH authentication
Network Countermeasures	0	Need more information
Attack Severity	2	

Severity score = (Criticality + Lethality) – (System + Net Countermeasures)

9. Defensive recommendation

The defenses are fine, but it is recommended to install a perimeter device (access router or firewall) to implement basic access control.

10. Multiple choice test question

This detect was an example of

a) Active targeting

b) Buffer overflow exploit

c) Stealth scans

d) SYN scans

Answer: D.

Detect 5 – RPC Probes

```
PART 1
Dec  1 20:10:21 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:26 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:31 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:36 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:41 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:46 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:10:51 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
```

```
Dec  1 20:10:56 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:11:01 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:11:06 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:11:11 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)
Dec  1 20:11:16 hostre rpcbind: refused connect from 65.211.149.171 to
getport(status)

PART 2
Dec  1 20:13:37 hostj snort: RPC portmap request rstatd [Classification:
Attempted Information Leak   Priority: 3]: 65.211.149.171:727 -> z.y.x.66:111
Dec  1 20:13:37 hosty snort: [ID 702911 auth.alert] [1:583:1] RPC portmap
request rstatd [Classification: Attempted Information Leak] [Priority: 3]:
{UDP} 65.211.149.171:726 -> z.y.x.34:111
```

1. Source of Trace:

The traces were collected from incidents.org:
http://www.incidents.org/archives/intrusions/msg02719.html

2. Detect was generated by:

The first part of this detect was collected from Unix message logs with the following format:

[Date & time][Hostname][Daemon/program][Process id][Messages]

The second part of this detect was generated by Snort IDS, which sent the alert messages to syslog in the following format:

[Date & Time] [Hostname] Snort: [Generator ID: Snort ID: Revision number] [Message] [Priority] {Protocol} [Source IP: port -> destination IP: port]

This detect was generated between Dec 1 20:10:21 and Dec 1 20:13:37.

3. Probability the source address was spoofed:

Low. The attacker attempted to initiate a connection to the victim host, and had to reply on the reply packets from it.

4. Description of attack:

The attacker tried to directly access to the RPC service by sending getport () function of portmapper. The attacker sent the getport () to the host every five seconds; therefore, it could be an automated tool for portmap scanning.

5. Attack mechanism:

Remote Procedure Calls (RPCs) are on the top ten most-exploitable-vulnerability list posted at SANS website. They are a programming interface for a client/server relationship between two systems. Basically, the client starts a process or procedure across network, and the server uses ephemeral ports for these services or procedures. A program called portmap listens on port 111 (TCP and UDP) and provides the program number of the service. There are four portmap functions:

- Set: used to register program number, version number and protocol.

- Unset: used to remove the existing port mapping.
- Getport: used by RPC client to obtain protocol, version number and port of a given program number.
- Dump: reply queries about the records in the portmapper database. (rpcinfo)

In this detect, the attacker used getport (status) rpc request to get the version number, protocol and port number of status program. If the rpc request succeeded, the attacker could've use the protocol and port number information to connect to the victim host.

6. Correlations:

Similar detect can be found in the following website:

http://www.sans.org/y2k/040901-1500.htm

http://www.incidents.org/archives/intrusions/msg01211.html

http://www.incidents.org/archives/intrusions/msg02130.html

7. Evidence of active targeting:

This detect could be an evidence of active targeting. The attacker sent rpc request to the victim host every five seconds, however, in part 2 of this detect, the attacker started probing other hosts after two minutes. That can be explained the attacker would change attack target if it got no response from the former one.

8. Severity

Component	Score	Comment
Target Criticality	3	Unix NFS server
Attack Lethality	4	RPC probing
System Countermeasures	3	Portmapper is not vulnerable
Network Countermeasures	3	Snort IDS
Attack Severity	1	

Severity score = (Criticality + Lethality) – (System + Net Countermeasures)

9. Defensive recommendation

It is recommended to disable RPC service on the hosts if it is not needed. Also block RPC service ports (portmapper 111 + ephemeral ports >32771) from external network using a firewall or router.

10. Multiple choice test question:

Which of the following portmapper function provides the response to the program rpcinfo?

a) set

b) unset

c) getport

d) dump

Answer: D

Assignment 3: Analysis This

Overview

The goal of this log analysis report is to provide a comprehensive security monitoring for GIAC University. Based on the Snort IDS Alert, Portscan and Out of Spec logs, we will provide insight analysis, detect the possibly security concerns and review current security policy. Finally, we will provide some defensive recommendations.

The following analysis is based on the logs that were received from the current Snort IDS setup in GIAC University. All the numbers are based on the logs from Sunday October 14 0:00:00 to Thursday October 18 23:59:59 EDT.

Traffic Analysis

A list of detected signatures and number of occurrences are shown below. We will analyze these signatures base on top 5 source hosts, top 5 destination hosts, signature information, correlation and defensive recommendation.

The following table shows the top 20 Snort signatures during the log period:

Event Signature	#
WEB-MISC Attempt to execute cmd	37154
MISC Large UDP Packet	36112
ICMP Echo Request speedera	26964
spp_http_decode	20356
IDS552/web-iis_IIS ISAPI Overflow ida nosize	13155
INFO MSN IM Chat data	8320
WEB-MISC prefix-get //	8135
Port 55850 tcp - Possible myserver activity - ref. 010313-1	7972
ICMP Echo Request Nmap or HPING2	7639
Watchlist 000220 IL-ISDNNET-990517	4808
ICMP Destination Unreachable (Communication Administratively Prohibited)	4519
MISC source port 53 to <1024	4482
CS WEBSERVER - external web traffic	4044
WEB-MISC 403 Forbidden	3711
MISC traceroute	3599
SMB Name Wildcard	3494
Possible trojan server activity	2994
Incomplete Packet Fragments Discarded	2823
TFTP - Internal TCP connection to external tftp server	1822
SMTP relaying denied	1422

We will provide detail analysis of the main signatures:

WEB-MISC Attempt to execute cmd

Top 5 Source host

Address	#
212.29.222.114	16974
212.29.222.118	855
213.10.162.83	852
63.125.75.163	604
193.117.39.80	579

Top 5 Destination host

Address	#
MY.NET.201.218	301
MY.NET.204.197	300
MY.NET.204.59	294
MY.NET.205.177	293
MY.NET.207.175	284

Signature information

The detect was generated by the following Snort rule:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-MISC Attempt to
execute cmd"; flags: A+; content:"cmd.exe"; nocase;)
```

Based on this rule, any http request that contains “cmd.exe” will trigger this signature. The attackers tried to exploit the Microsoft IIS vulnerabilities on target hosts. Possible attacks including IIS Unicode attack, IIS extended Unicode directory transversal.

Defensive Recommendation

If the target hosts are installed with Microsoft IIS, always apply the latest patches.

It is also recommended to block these specific attackers, since they generated the majority of noises.

Correlation

Please refer to assignment 2 detect 1 – Nimda Worm

Also, the similar attack can be found in the following website:
http://www.incidents.org/archives/intrusions/msg01539.html

MISC Large UDP Packet

Top 5 Source host

Address	#
61.150.5.19	13027
61.134.9.88	7398
209.190.237.123	4172
61.153.17.210	2022
211.91.255.142	1473

Top 5 Destination host

Address	#
MY.NET.111.221	12684
MY.NET.111.142	5968
MY.NET.70.134	4172
MY.NET.112.244	2526
MY.NET.109.74	1773

Signature information

```
10/14-00:18:17.502380  @ MISC Large UDP Packet @ 64.152.4.136:0 @
MY.NET.241.210:0
10/14-00:18:19.180932  @ MISC Large UDP Packet @ 64.152.4.136:0 @
MY.NET.241.210:0
10/14-00:18:19.623191  @ MISC Large UDP Packet @ 64.152.4.136:0 @
MY.NET.241.210:0
```

The large UDP packets are very suspicious because the external attackers connect to internal host through port 0. There are 28705 “ MISC large UDP packet” detects are from port 0 to port 0, 398 detects from high ports to port 1681.

Defensive Recommendation

Monitor these source hosts carefully. Check the internal hosts if they are infected by any Virus/Trojan/Worm. Install anti-virus software to provide better detection.

Correlation

http://www.sans.org/y2k/092400.htm

ICMP Echo Request speedera

Source host

One source host MY.NET.205.130 was collected.

Destination host

Address	#
216.102.120.11	12701
24.70.191.95	12255
24.39.129.167	2008

Three destination hosts were collected.

Signature information

```
10/14-17:16:49.880613  @ ICMP Echo Request speedera @ MY.NET.205.130 @
24.39.129.167
10/14-17:16:50.045545  @ ICMP Echo Request speedera @ MY.NET.205.130 @
24.39.129.167
```

Speedera is a company that offers global web load balancing. Speedera's Global traffic Management (GTM) system places copies of customer site throughout the world. When users want to access the customer site, they are routed to the closest web site hosting your pages, in order to reduce network access time, while providing full redundancy.

Defensive Recommendation

Firewall or access router should block icmp traffic into internal network. Make sure all activity to internal DNS servers is logged to ensure you are able to detect malicious activity. To save network bandwidth, be sure to block the ICMP flooding into internal network.

Correlation

http://www.sans.org/y2k/practical/Faud_Khan_GCIA.doc

spp_http_decode

Top 5 Source host

Address	#
212.29.222.114	9162
213.10.162.83	852
212.29.222.118	425
63.125.75.163	328
62.110.19.12	298

Top 5 Destination host

Address	#
MY.NET.202.75	171
MY.NET.201.218	171
MY.NET.201.199	167
MY.NET.205.177	163
MY.NET.207.175	153

Signature information

It is another IIS vulnerabilities Snort signature. If someone tried to use Unicode attack against the system, it would trigger the following rules:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80: (msg:"spp_http_decode: IIS
Unicode attack detected"; flags: A+; content: "%c0";)
alert tcp $EXTERNAL_NET any -> $HOME_NET 80: (msg:"spp_http_decode: IIS
Unicode attack detected"; flags: A+; content: "%af";)
……
(skip)
```

Defensive Recommendation

If Microsoft IIS in installed on the systems, be sure to install the latest patches. Also, apply more strict access control list on perimeter devices. (Firewall or router)

Correlation

http://www.incidents.org/archives/intrusions/msg00797.html

IDS552/web-iis_IIS ISAPI Overflow ida nosize

Top 5 Source host

Address	#
65.106.101.135	69
213.38.196.136	44
65.70.65.126	42
212.3.152.20	33
212.150.70.65	32

Top 5 Destination host

Address	#
MY.NET.82.96	6
MY.NET.7.231	5
MY.NET.25.195	5
MY.NET.191.71	5
MY.NET.185.244	5

Signature information

```
10/14-00:09:25.314982  @ IDS552/web-iis_IIS ISAPI Overflow ida nosize @
61.217.63.144:4120 @ MY.NET.203.122:80
10/14-00:11:18.851796  @ IDS552/web-iis_IIS ISAPI Overflow ida nosize @
129.105.243.43:3386 @ MY.NET.70.201:80
10/14-00:13:56.333271  @ IDS552/web-iis_IIS ISAPI Overflow ida nosize @
62.244.0.43:36072 @ MY.NET.69.219:80
```

IIS installs several ISAPI extensions - .dlls that provide extended functionality. Among these is idq.dll, which is a component of Index Server (known in Windows

2000 as Indexing Service) and provides support for administrative scripts (.ida files) and Internet Data Queries (.idq files).

The idq.dll extension contains a buffer overflow in the code handling input URLs. Idq.dll runs in the System context, so exploiting the vulnerability would give the attacker complete control of the server.

Defensive Recommendation

Apply the latest Microsoft IIS patches. Hardening the operating system of the web server.

Correlation

CVE CAN-2001-0500

http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2001-0500

Port 55850 tcp - Possible myserver activity - ref. 010313-1

Top 5 Source host

Address	#
MY.NET.226.58	4538
194.213.87.193	2488
MY.NET.214.82	780
24.40.0.139	84
207.232.184.100	29

Top 5 Destination host

Address	#
194.213.87.193	4538
MY.NET.226.58	2488
24.40.0.139	780
MY.NET.214.82	84
MY.NET.223.154	29

Signature information

```
10/14-01:30:23.952472  @ Port 55850 tcp - Possible myserver activity - ref.
010313-1 @ MY.NET.214.82:55850 @ 24.40.0.139:4320
10/14-01:30:24.028020  @ Port 55850 tcp - Possible myserver activity - ref.
010313-1 @ MY.NET.214.82:55850 @ 24.40.0.139:4320
```

Myserver is a DDOS agent.
It binds to UDP 55850, and the rootkit installs trojans of ls and ps, so users won't see it running. The only way to see it is using netstat –an, find the port 55850. The rootkit and ddos tools are stored in "/lib/ ". Based on the log, we know MY.NET.214.82, 194.213.87.193, 207.232.184.100 and several internal and external hosts all listened on UDP port 55850, which means the rootkit and ddos tools were installed on those machines already. Therefore, the compromised internal machines are:

```
MY.NET.5.74
MY.NET.253.52
MY.NET.218.254
MY.NET.253.51
```

```
MY.NET.214.82
```

Defensive Recommendation

Remove the rootkit directory /lib/. Install anti-virus software to make sure the systems are not infected by any virus/worm/Trojan. Install anti-virus program to provide better detection.

Correlation

http://www.sans.org/y2k/082200.htm

Possible Trojan server activity

Top 5 Source host

Address	#
MY.NET.97.134	1080
MY.NET.223.74	891
206.48.84.109	461
199.8.18.253	215
MY.NET.98.172	139

Top 5 Destination host

Address	#
206.48.84.109	891
MY.NET.223.74	461
MY.NET.241.70	215
199.8.18.253	97
MY.NET.97.134	24

Signature information

```
10/15-04:47:13.914723  @ Possible trojan server activity @193.251.37.76:27374
@ MY.NET.220.22:1214
10/15-04:47:17.359019  @ Possible trojan server activity @ MY.NET.220.22:1214
@193.251.37.76:27374
```

This signature is triggered when attacker connect to the common Trojan port.

We found there are 2281 connection attempts to port 27374 (SubSeven: a Trojan that allows intruder control a system remotely), and 461 connections attempts to port 6346 (Gnutella: a peer-to-peer information sharing tool).

Based on the logs, the possible compromised hosts including 206.48.84.109, 199.8.18.253, MY.NET.210.222 and 128.121.193.86. All of them listen on port 27374

Defensive Recommendation

Install an anti-virus program to protect the system from these Trojans. For SubSeven Trojan, it may be required to remove files listed below manually:

```
WINDOWS\
DATA2.EXE
KERNEL16.DLL
NODLL.EXE
RUNDLL16.COM
SERVER.EXE
```

```
SYSTEMTRAYICON!.EXE
TINURAK.EXE
WINDOW.EXE
WINDOWS\SYSTEM\
LMDRKI_33.DLL
WATCHING.DLL
```

Correlation

SubSeven traces. http://www.sans.org/y2k/032800-2000.htm

Gnutella traces: http://www.sans.org/y2k/052100.htm

The Trojan port list: http://www.sans.org/newlook/resources/IDFAQ/oddports.htm

Top Talkers List

We summarized the most active hosts from Snort IDS logs:

Top 10 Alert Source Addresses

We define top alert source addresses based on the number of alert events they generated in the logs.

Address	#
MY.NET.205.130	26964
212.29.222.114	26160
61.150.5.19	13031
61.134.9.88	8868
MY.NET.225.6	5502
MY.NET.226.58	4542
209.190.237.123	4174
MY.NET.14.1	3864
MY.NET.100.165	3204
194.213.87.193	2488
61.153.17.210	2022

Top 10 Alert Destination Addresses

We define top alert destination addresses based on the number of alert events they generated in the logs.

Address	#
216.102.120.11	12701
MY.NET.111.221	12692
24.70.191.95	12255
MY.NET.253.114	7962
MY.NET.111.142	5976
194.213.87.193	4539
MY.NET.100.165	4239
MY.NET.70.134	4174
MY.NET.140.9	4027
MY.NET.112.244	2528

Top 10 Alert Destination Port

Port #	#
80(http)	85695
0	32128
1863(MSN Messenger)	5910
55850(myserver ddos tool)	4664
53(dns)	4490
6346(Gnutella)	3589
137(NetBIOS)	3496
27374(SubSeven)	2281
69(bootp)	1405
21(ftp)	1108

From the top 10 destination port statistics, port 80 (web attacks/scans) is still the majority in all scans. Port 1863 (MSN Messenger) and 6346 (Gnutella) are not malicious but could have some potential security breaches. Port scans on port 53, 137, 21, 55850 and 27374 are reconnaissance activities and could be the prelude of future attacks.

Top 5 External Source Hosts

We will provide individual analysis for top 5 external source hosts:

Address	#
212.29.222.114	26964
61.150.5.19	13031
61.134.9.88	8868
209.190.237.123	4174
194.213.87.193	2488

212.29.222.114

Whois information:

```
inetnum:      212.29.222.96 - 212.29.222.127
netname:      EFRAT-1
descr:        Efrat
country:      IL
```

```
admin-c:        OH624-RIPE
tech-c:         DB1523-RIPE
status:         ASSIGNED PA
mnt-by:         RIPE-NCC-NONE-MNT
changed:        dbenjamin@barakitc.co.il 19981018
source:         RIPE
```

Signatures generated from 212.29.222.114:

- 16974 WEB-MISC Attempt to execute cmd
- 9162 spp_http_decode: IIS Unicode attack detected
- 28 ICMP Destination Unreachable (Network Unreachable)
- 8 High port 65535 tcp - possible Red Worm - traffic
- 6 Possible trojan server activity
- 6 Port 55850 tcp - Possible myserver activity - ref. 010313-1
- 4 WEB-IIS Unauthorized IP Access Attempt
- 1 WEB-MISC 403 Forbidden
- 1 CS WEBSERVER - external web traffic

This attacker tried to exploit IIS-related vulnerabilities aggressively, including IIS Unicode attack; IIS extended Unicode directory transversal vulnerabilities, Red Worm, Trojans…etc

We suggest monitoring this host carefully and using a firewall to block unnecessary services.

As for web servers, always apply the latest patches and disable unnecessary services.

61.150.5.19

Whois information:

```
inetnum             61.150.0.0 - 61.150.31.255
netname             SNXIAN
descr               xi'an data branch,XIAN CITY
SHAANXI PROVINCE
country             CN
admin-c             WWN1-AP, inverse
tech-c              WWN1-AP, inverse
mnt-by              MAINT-CHINANET-SHAANXI, inverse
mnt-lower           MAINT-CN-SNXIAN, inverse
changed             ipadm@public.xa.sn.cn 20010309
source              APNIC
```

Signatures generated from 61.150.5.19:

- 13027 MISC Large UDP Packet
- 75 ICMP Fragment Reassembly Time Exceeded
- 4 High port 65535 udp - possible Red Worm - traffic

The log shows the attacker sent lots of UDP packets to MY.NET.111.221 on multiple ports. It indicates either the attacker kept doing reconnaissance activities or the host MY.NET.111.221 has been compromised!

It is recommended to pay special attention to the host MY.NET.111.221. Check if it is infected by any Virus/Trojan/Worm. And block the unnecessary UDP ports by using a firewall or access router.

61.134.9.88

Whois information:

```
inetnum          61.134.3.0 - 61.134.20.95
netname          SNXIAN
descr            XI'AN DATA BUREAU
country          CN
admin-c          WWN1-AP, inverse
tech-c           WWN1-AP, inverse
mnt-by           MAINT-CHINANET-SHAANXI, inverse
mnt-lower        MAINT-CN-SNXIAN, inverse
changed          ipadm@public.xa.sn.cn 20010427
source           APNIC
```

Signatures generated from 61.134.9.88:

- 7398 MISC Large UDP Packet
- 1467 Incomplete Packet Fragments Discarded
- 32 ICMP Fragment Reassembly Time Exceeded
- 3 High port 65535 udp - possible Red Worm - traffic

In the log, 61.134.9.88 sent UDP packets to several internal hosts: MY.NET.111.142 MY.NET.112.244, MY.NET.153.147, MY.NET.153.203 and MY.NET.111.145. The abnormal point is the UDP source and destination port number are both set to 0! Normal traffic shouldn't have the source or destination port set to 0, so the host 61.134.9.88 is extremely suspicious for attacking the internal hosts.

It is recommended to check on those internal hosts, look at the syslog or event viewer for any suspicious activities. If necessary, reinstall the systems and install IDS or anti-virus software on these machines to provide better protection to these systems.

209.190.237.123

Whois information:

```
Atlantech Online, Inc. (NETBLK-AOI1999B)
1010 Wayne Avenue, Suite 630
Silver Spring, MD 20910
US

Netname: AOI1999B
```

```
Netblock: 209.190.192.0 - 209.190.255.255
Maintainer: ATON
```

Signatures generated by 209.190.237.123:

- 4172 MISC Large UDP Packet
- 2 ICMP Fragment Reassembly Time Exceeded
- 2 High port 65535 udp - possible Red Worm – traffic

For traffic initiated from 209.190.237.123, we found again for those large UDP packets, their source and destination ports are set to 0. This attacker sent large amount of UDP packets to MY.NET.70.134 targeting port 0. It is possible that MY.NET.70.134 has been compromised. Please check the machine if it is infected by any Virus/Trojan/Worm. If necessary, reinstall the systems and install IDS or anti-virus software on these machines to provide better protection to these systems.

194.213.87.193

Whois information:

```
inetnum:      194.213.87.0 - 194.213.87.255
netname:      SE-OM
descr:        OM gruppen
country:      SE
admin-c:      NL9-RIPE
tech-c:       NL9-RIPE
rev-srv:      omwld.omgroup.com
rev-srv:      dns.telenordia.se
rev-srv:      ns0.pipex.net
status:       ASSIGNED PA
notify:       postmaster@telenordia.se
mnt-by:       AS5556-MNT
changed:      gunilla.malmqvist@telenordia.se 19970813
source:       RIPE
```

Signatures generated by 194.213.87.193:

- 7026 Port 55850 tcp - Possible myserver activity - ref. 010313-1
- 1 INFO Inbound GNUTella Connect accept

The log shows there is a UDP connection between 194.213.87.193 and MY.NET.226.58 using port 6346 and 55850.

```
10/15-02:52:45.221533  [**] Port 55850 tcp - Possible myserver activity -
ref. 010313-1 [**] 194.213.87.193:55850 -> MY.NET.226.58:6346
10/15-02:52:46.676949  [**] Port 55850 tcp - Possible myserver activity -
ref. 010313-1 [**] MY.NET.226.58:6346 -> 194.213.87.193:55850
```

It seems the DDOS agent myserver is installed on 194.213.87.193.
The DDOS agent binds to UDP 55850, and the rootkit installs Trojans of ls and ps, so users won't see it running. The only way to see it is using 'netstat –an', find the port 55850. The rootkit and ddos tools are stored in "/lib/ ".

It is possible the host 194.213.87.193 is compromised by MY.NET 226.58; please investigate MY.NET.226.58.

Portscan Logs

Top 10 Scanning Hosts

Host	#	Traffic Categorization
MY.NET.160.114	256831	Many different hosts and ports
205.188.233.121	18961	UDP Port 6970 and 6972 (RealAudio)
205.188.244.57	16861	UDP Port 6970 and 6972 (RealAudio)
205.188.233.153	16842	UDP Port 6970 and 6972 (RealAudio)
205.188.233.185	15062	UDP Port 6970 and 6972 (RealAudio)
205.188.244.121	13260	UDP Port 6970 (RealAudio)
205.188.246.121	13089	UDP Port 6970 (RealAudio)
MY.NET.221.250	11308	Many different hosts and ports
MY.NET.179.74	3337	UDP 6970, 6971 (Real Audio), UDP 1041
MY.NET.201.150	3191	6112, 64771, 16774, 1214, 43987

From the top 10 scanning hosts and the traffic categorization, we can summarize the following:

The majority of traffic from external network was Real Audio related or game-related. The whois database of 205.188.0.0 network:

```
Netname: AOL-DTC
Netblock: 205.188.0.0 - 205.188.255.255

Coordinator:
America Online, Inc.  (AOL-NOC-ARIN)  domains@AOL.NET
703-265-4670
```

MY.NET.160.114 and MY.NET.221.150 scanned for many hosts and ports. We suggest monitoring these two hosts carefully.

MY.NET.201.150 was logged for 3191 events. However, most of the scanning traffic from this host targeted on unknown high ports. I cannot find port information about 6112, 64771, 16774, 1214 and 43987 on the web. My guess is that either new application or bad hardware causes the scanning on these ports.

Top 10 Destination Hosts:

Host	#	Traffic Categorization
MY.NET.184.23	6958	6970,6972(RealAudio), 21(ftp), 53(dns)
131.204.205.119	6896	27005, 1619, 1163, 1150
MY.NET.145.166	6527	6970, 6972(RealAudio)
MY.NET.108.15	6522	6970(RealAudio), 21(ftp), 53(dns)
MY.NET.109.62	6347	6970(RealAudio)
MY.NET.110.33	6307	6970(RealAudio)
MY.NET.145.197	6243	6970(RealAudio)
4.42.50.83	4444	27005, 2165,1980
24.76.42.84	4219	27005
MY.NET.151.85	3871	6970, 1214,1

From the top 10 destination hosts, we can still find lots of real Audio streaming connections, and Internet gaming traffic. To see the port list of Internet Game,

please refer to Matt Scarborough's article: What are the Signs of Internet Gaming? At: http://www.incidents.org/detect/gaming.php

The three external hosts 131.204.205.119 is from Auburn University, 24.76.42.84 is from Canada and the host 4.42.50.83 is from an ISP – Genuity.

Out of Spec Logs

From the out of spec logs we can summarize the following:

1. 205.233.33.61 was doing an ftp scan on internal network using scanning tools such as Synscan and Sscan2k-pre6.

```
10/14-21:14:04.800047 205.233.33.61:21 -> MY.NET.141.187:21
TCP TTL:25 TOS:0x0 ID:39426
**SF**** Seq: 0x20F334ED   Ack: 0x1C43D9E6   Win: 0x404
00 00 00 00 00 00                                  ......
```

2. 199.183.24.194 was doing a slow scan to internal network. It sent packets to multiple hosts on multiple ports.
3. Large amount of Gnutella traffic establish the connection between internal hosts and external hosts.

Defensive Recommendation

In summary, we provide the following defensive recommendation:

1. Apply the latest patches routinely on all operating system, web servers, ftp servers…etc. If possible, the internal hosts should be installed host based IDS systems such as Tripwire and Portsentry.
2. Installation and maintenance of a more restrictive firewall policy and router ACL.
3. Anti-virus software should be installed on all desktop computers.
4. Analysis the following possible compromised hosts:
   MY.NET.111.221, MY.NET.111.142, MY.NET.112.244, MY.NET.153.147, MY.NET.153.203, MY.NET.111.145, MY.NET.70.134. MY.NET.111.142, MY.NET.112.244, MY.NET.153.147, MY.NET.153.203, MY.NET.111.145, MY.NET.210.222, MY.NET.226.58.
5. Perform regular audits on your network using scanning tools.
   Nessus (http://www.nessus.org) is a good example of the scanning tools.

Analysis Process

I downloaded the following files from the assignment data site:

Alert.011014.gz	oos_Oct.14.2001.gz	scans.011014.gz
Alert.011015.gz	oos_Oct.15.2001.gz	scans.011015.gz
Alert.011016.gz	oos_Oct.16.2001.gz	scans.011016.gz
Alert.011017.gz	oos_Oct.17.2001.gz	scans.011017.gz
Alert.011018.gz	oos_Oct.18.2001.gz	scans.011018.gz

Alert logs analysis

After putting all alert files in one file, I used snort_stat.pl (http://xanadu.incident.org/snort/) to generate some statistic information of the logs.

Because snort_stat.pl doesn't handle the string "MY.NET", I have to replace "MY.NET" with some numbers to run it. I choose "10.222" to replace "MY.NET"

However, I still need more information, so I did some data manipulation on the alert file. First I filter "spp_portscan" out of alert file.

```
grep –v spp_portscan alert > alert1
```

Then replace [**] and -> with @ (@ is my delimiter)

```
:%s/\[**\]/@/g
:%s/->/@/g
```

The alert log format now looks like below:

```
10/14-00:00:44.788518  @ INFO MSN IM Chat data @ MY.NET.209.14:1879 @
64.4.12.161:1863
```

And we can retrieve any statistical information by using grep, awk, cut, sort, uniq commands. For example, to get statistic information about destination port:

```
cut -f4 -d@ alert | awk -F":" '{print $2}'| sort | uniq -c | sort -rn >
all_dst_port
```

Portscan logs analysis

In order to process the Snort portscan logs into a comma delimited format, the following UNIX commands were used. The following import filter was used to extract source ip, source port, destination ip and destination port information into a comma delimited text file.

```
cat snort_scan | awk '{print $4, $6, $7}' | tr : ',' | tr ' ' , > scan
```

Each line in the scan file has the format: src ip, src port, dst ip, dst port, protocol

```
195.98.63.11,1949,MY.NET.226.18,2084,UDP
```

Then, to get the source ip statistic information, use the following command:

```
cut –f1 -d, scan |sort | uniq -c | sort -rn
```

The same command can be used to get statistical information about destination ip, port…. etc.

References

Northcutt, Stephen and Judy Novak. Network Intrusion Detection, An Analysts Handbook, 2nd Edition. Indianapolis, IN: New Riders Publishing, 2000.

Stephen Northcutt, Mark Cooper, Matt Fearnow, Karen Frederick. Intrusion Signatures and Analysis, Indianapolis, IN: New Riders Publishing, 2001.

W. Richard Stevens. TCP/IP Illustrated, Volume 1. Boston: Addison-Wesley, 1994.

Correlation sources:

http://cve.mitre.org

http://www.incidents.org

http://www.securityfocus.com

http://www.google.com

GCIA Practical Assignments:

Lenny Zelster. http://www.sans.org/y2k/practical/Lenny_Zeltser.htm

Guy Bruneau. http://www.sans.org/y2k/practical/Guy_Bruneau.doc

Teri Bidwell. http://www.sans.org/y2k/practical/Teri_Bidwell_GCIA.doc

Faud Khan. http://www.sans.org/y2k/practical/Faud_Khan_GCIA.doc

Donald Smith. http://www.giac.org/practical/donald_smith_gcia.doc

Last Updated: September 22nd, 2017

Upcoming Training

SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Berlin 2017	Berlin, Germany	Oct 23, 2017 - Oct 28, 2017	Live Event
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Paris November 2017	Paris, France	Nov 13, 2017 - Nov 18, 2017	Live Event
Community SANS Pensacola SEC503	Pensacola, FL	Nov 27, 2017 - Dec 02, 2017	Community SANS
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Munich December 2017	Munich, Germany	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LA	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Las Vegas 2018	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	Live Event
SANS London February 2018	London, United Kingdom	Feb 05, 2018 - Feb 10, 2018	Live Event
SANS Dallas 2018	Dallas, TX	Feb 19, 2018 - Feb 24, 2018	Live Event
SANS Northern VA Spring - Tysons 2018	Tysons, VA	Mar 17, 2018 - Mar 24, 2018	Live Event
SANS Secure Canberra 2018	Canberra, Australia	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS 2018	Orlando, FL	Apr 03, 2018 - Apr 10, 2018	Live Event
SANS Baltimore Spring 2018	Baltimore, MD	Apr 21, 2018 - Apr 28, 2018	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced