Advanced Security Reporting With Nessus

Christopher McCafferty
GIAC Security Essentials Certification (GSEC)
Practical Assignment
Version 1.4 – Option 1
July 23, 2002

Table of Contents

1 Abstract

For penetration testers working on a consulting basis, clear and concise test reports are important. Typically, penetration test reports prepared by the consultant are read by people at several levels in the client company. For that reason, the penetration test reports should be simple enough for client company decision-makers to understand their security posture, yet detailed enough to allow the client IT staff to understand the security problems on a technical level. The goal of this paper is to describe techniques for extending and automating penetration test reports using Nessus.

2 Introduction

The goal of this project is to present a method for creating professional and useful penetration test reports from Nessus for clients in a short amount of time. Nessus is a powerful open-source penetration testing tool whose development is managed by Renaud Deraison[1]. Chan Tuck Wai describes a straightforward method for penetration testing using Nessus in his SANS article[2]. The GSEC Security Essentials Toolkit explains how to set up Linux, and how to compile and run Nessus[3]. The Nessus client can prepare penetration test reports in a variety of formats:

- **.NBE format** – This format is the native format for Nessus. NBE results can be saved and opened by the Nessus client.
- **LaTex format** – This format can be converted to PDF files, but is not customizable.
- **ASCII text format** – This format is a simple text output of results.
- **HTML with Pies and Graphs** – This format is a popular one because it presents the results in a series of web pages that can be emailed to or printed for the client company. The index page also contains graphs that illustrate the results. However, there are some drawbacks to this format. It is not customizable – it is always the same format and does not use Cascading Style Sheets (CSS)[4], which would allow authors to change the look and feel of this format. The graphs could look better – they are too "grainy" for a professional-looking report. Also, the results are not sorted – I have had complaints from clients who cannot search through the information effectively. Finally, printing is tedious, as the results sometimes contain hundreds of web pages.
- **XML format** – This format is currently the most promising, because it contains more information about the penetration test than the .NBE format, such as elapsed time for the test and the Nessus plugins that were used during the test. Unfortunately, the XML is not well-formed enough (as of Nessus version 1.2.3) to use in reports. I verified this using the Microsoft XML validation tool[5]. But, if problems with this format are resolved in future Nessus versions, XML will probably become the preferred format for several reasons. XML is directly accessible to report-writing tools like Seagate Crystal Reports. Also, XML is becoming the de-facto standard for data exchange. Moreover, XML is easier to translate and format to a variety of mediums.

Using the methods described in this paper, you can create virtually any type of report from Nessus test results. The same techniques can be used with the XML format output once its problems have been resolved.

3 Setup

This section lists the necessary software and data you will need to get started.

3.1 Software

In order to generate the reports described below, I used the following:

- Microsoft Windows 2000 or XP
- Microsoft Excel XP

- Microsoft Excel Web Query Feature
- Microsoft Access XP
- RetHat Linux 7.3[6]
- NMap[7]
- Nessus[1]

3.2 The Nessus NBE File

The first thing that you will need is a Nessus results file. I have chosen the .NBE format, because it is a simple delimited text file containing all the raw results of the penetration test. NBE is the native format for raw Nessus results in the current version (1.2.3) of Nessus. To obtain Nessus results, you must compile and run a penetration test on a number of hosts and save the results as NBE. The NBE file will be imported into a Microsoft Access table for use in the report.

3.3 Nessus Plugins Table

For the purposes of reporting, I have found it very useful to maintain a table of all existing Nessus plugins. Instead of recording all the plugins myself, I import the list from www.nessus.org. Here is the procedure for importing the lists:

1. Create an Excel spreadsheet.
2. Save and name the file "NessusPlugins.xls".
3. On the first worksheet, go to Data | Import external data | New web query.
4. For the query location, enter http://cgi.nessus.org/plugins/dump.php3.
5. Select the main table to import.

Note: After running the web query, it is necessary to manually enter the Nessus plugin ID for each plugin in an adjacent Excel column. Without the plugin ID column, it will be impossible to generate really useful reports. Initially, this is a lot of work, because there are about 1000 plugins at this time. But, the resulting reports are so excellent, that it will be worth your time to do this.

When you have imported the data and added the plugin ID's, your Excel spreadsheet should appear as shown below.

Test ID	Name	Family	Summary	Port(s)
10005	NetSphere	Backdoors	Checks for the presence of NetSphere	30100
10006	PC Anywhere	Backdoors	Checks for the presence PC Anywhere	
10794	PC Anywhere TCP	Backdoors	Checks for the presence PC Anywhere	5631 65301
	BackOrifice	Backdoors	Determines the presence of BackOrifice	

3.4 Reporting Database

Once you have prepared the previous items, you can create the reporting database.

1. Create a Microsoft Access database named “NessusReports.mdb”.
2. Open the database.
3. Import the Microsoft Excel worksheet that you created into a new Access table named “Plugins”.
4. Import the Nessus .NBE results file into a new Access table named “Results”. Note: the NBE formatted file is delimited with the “|” symbol. The necessary fields are: Host Name, Port, Test ID, Severity, and Information.

The “Plugins” table should look as shown below.

Test ID	Name	Family	Summary
10328	Default accounts	Misc.	Telnet to the remote host and try login/passwords
10330	Services	Misc.	Find what is listening on which port
10332	ftp writeable directories	FTP	checks if the remote FTP server has any world writeable dirs
10909	Brute force login (Hydra)	Misc.	Accounts brute force

The “Results” table should look as shown below.

Host Name	Port	Test ID	Severity	Information
svr1.foo.bar	unknown (1755/tcp)			
svr1.foo.bar	general/udp			
svr1.foo.bar	http (80/tcp)	10064	Security Hole	The Excite for Webservers is installed. This CGI has\na well known security flaw that lets anyone execute arbitrary\ncommands with the privileges of the http daemon (root or nobody).\n\nVersions newer than 1.1. are patched.\n\n\nSolution : if you are running version 1.1 or older, then\nupgrade it.\n\nRisk factor : Serious\nCVE : CVE-1999-0279\n\n

4 Mining the Data

Now that the preparation work is done, you can start writing some powerful queries. We will start by cleaning up the “Results” table. I wrote these queries, using the techniques described in the Microsoft Access Bible[8]. Another good resource is on the Microsoft Office web site[9].

4.1 Query: CleanResults

The CleanResults select query is used to smooth out some text anomalies in the NBE results file. Here is the SQL syntax for the query:

```
SELECT Results.[Host Name], Results.Port, Results.[Test ID], Results.Severity,
Results.Information, Nz([Information],"This port appears to be open.") AS
InfoClNz, Replace([InfoClNz],"\n"," ") AS InfoCleaned, Nz([Severity],"Other") AS
SevClean
FROM Results
WHERE (((Results.[Host Name]) Is Not Null))
ORDER BY Results.[Host Name], Results.Port;
```

The resulting query will remove some zero-value lines and remove persistent “\n” characters from the “Information” field. Otherwise, the query output is exactly the same as the “Results” table.

Host Name	Port	Test ID	Severity	Information	InfoCleaned	SevClean
mail.foo.bar	general/tcp	10336	Security Warning	Nmap only scanned 65000 TCP ports out of 65535.Nmap did not do a UDP scan, I guess.\n	Nmap only scanned 65000 TCP ports out of 65535.Nmap did not do a UDP scan, I guess.	Security Warning
mail.foo.bar	general/tcp	10336	Security Warning	Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98\n\n	Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98	Security Warning
mail.foo.bar	general/tcp	10879	Security Warning	The plugin port_shell_execution.nasl was too slow to finish - the server killed it\n\n	The plugin port_shell_execution.nasl was too slow to finish - the server killed it	Security Warning

4.2 Query: Hosts

The Hosts select query generates a simple list of the hosts that were scanned by Nessus:

```
SELECT Results.[Host Name]
FROM Results
GROUP BY Results.[Host Name]
HAVING (((Results.[Host Name]) Like "*"));
```

Host Name
mail.foo.bar
mktg.foo.bar
router.foo.bar
sales.foo.bar
svr1.foo.bar
svr3.foo.bar
vpn.foo.bar
vpn2.foo.bar

4.3 Query: HostCount

The HostCount select query generates an integer representing the number of hosts that were scanned by Nessus. This makes it possible to say, “12 hosts were scanned for this report”:

```
SELECT Count(Hosts.[Host Name]) AS [CountOfHost Name]
FROM Hosts;
```

CountOfHost Name
8

4.4 Query: CountIssues

The CountIssues select query totals the Security Warnings, Security Notes, and Security Holes.

```
SELECT [SevClean],Count(*) AS [Count] FROM [CleanResults]  GROUP BY
[SevClean];
```

SevClean	Count
Other	7
Security Hole	97
Security Note	66
Security Warning	59

4.5 Query: Top 5 Common Holes by Port

The Top 5 Common Holes by Port select query will display the top 5 ports with security holes, and the number of holes found for each port:

```
SELECT TOP 5 CleanResults.Port, CleanResults.Severity, Count(*) AS [Count]
FROM CleanResults
GROUP BY CleanResults.Port, CleanResults.Severity
HAVING (((CleanResults.Severity)="Security Hole"))
ORDER BY Count(*) DESC;
```

Port	Severity	Count
http (80/tcp)	Security Hole	54
https (443/tcp)	Security Hole	40
smtp (25/tcp)	Security Hole	3

4.6 Query: Top 5 Hosts with Holes

The Top 5 Hosts with Holes select query will display the top 5 hosts with security holes, and the number of holes found for each host:

```
SELECT TOP 5 CleanResults.[Host Name], Count(CleanResults.SevClean) AS
CountOfSevClean
FROM CleanResults
WHERE (((CleanResults.SevClean)="Security Hole"))
GROUP BY CleanResults.[Host Name]
ORDER BY Count(CleanResults.SevClean) DESC;
```

Host Name	CountOfSevClean
svr3.foo.bar	31
mail.foo.bar	28
svr1.foo.bar	24
vpn.foo.bar	14

4.7 Query: Top 5 Hosts with Severity

The Top 5 Hosts with Severity select query extends the results from the Top 5 Hosts with Holes query to include a count of security problems of varying severity for the critical hosts. This query is very useful for generating an effective graph:

```
SELECT [Top 5 Hosts with Holes].[Host Name], CleanResults.SevClean,
Count(*) AS [Count]
FROM [Top 5 Hosts with Holes] INNER JOIN CleanResults ON [Top 5 Hosts
with Holes].[Host Name] = CleanResults.[Host Name]
GROUP BY [Top 5 Hosts with Holes].[Host Name], CleanResults.SevClean;
```

Host Name	SevClean	Count
mail.foo.bar	Security Hole	28
mail.foo.bar	Security Note	17
mail.foo.bar	Security Warning	14
svr1.foo.bar	Other	2
svr1.foo.bar	Security Hole	24
svr1.foo.bar	Security Note	17
svr1.foo.bar	Security Warning	14
svr3.foo.bar	Other	2
svr3.foo.bar	Security Hole	31
svr3.foo.bar	Security Note	16
svr3.foo.bar	Security Warning	16
vpn.foo.bar	Other	1
vpn.foo.bar	Security Hole	14
vpn.foo.bar	Security Note	7
vpn.foo.bar	Security Warning	5

4.8 Query: Host Holes Totals

The Host Holes Totals select query counts the total security problems by the plugin ID family:

```
SELECT Plugins.Family, Count(*) AS [Count]
FROM CleanResults INNER JOIN Plugins ON CleanResults.[Test
ID]=Plugins.[Test ID]
GROUP BY Plugins.Family;
```

Family	Count
Backdoors	7
CGI abuses	129
Firewalls	4
Gain a shell remotely	3
General	36
Misc.	27
SMTP problems	3

4.9 Query: Host Holes by Type

The Host Holes by Type select query lists the hosts exhibiting particular security problems in an easily-referenced table:

```
SELECT Plugins.Family, Plugins.Name, Results.[Host Name], Count(*) AS
[Count]
```

```
FROM Results INNER JOIN Plugins ON Results.[Test ID] = Plugins.[Test ID]
GROUP BY Plugins.Family, Plugins.Name, Results.[Host Name];
```

Family	Name	Host Name	Count
Backdoors	Dansie Shopping Cart backdoor	mail.foo.bar	2
Backdoors	Dansie Shopping Cart backdoor	svr1.foo.bar	2
Backdoors	Dansie Shopping Cart backdoor	svr3.foo.bar	2
Backdoors	Dansie Shopping Cart backdoor	vpn.foo.bar	1
CGI abuses	bizdb1-search.cgi located	mail.foo.bar	2
CGI abuses	bizdb1-search.cgi located	svr1.foo.bar	2
CGI abuses	bizdb1-search.cgi located	svr3.foo.bar	2
CGI abuses	bizdb1-search.cgi located	vpn.foo.bar	1

4.10 Query: Host Severity Summary

The Host Severity Summary crosstab query produces a totals sheet of security problems, organized by host, port, and severity:

```
TRANSFORM Count(Results.InfoCleaned) AS CountOfInfoCleaned
SELECT Results.[Host Name], Results.Port, Count(Results.InfoCleaned) AS
[Total Of InfoCleaned]
FROM Results
GROUP BY Results.[Host Name], Results.Port
PIVOT Results.SevClean;
```

Host Name	Port	Total Of InfoCleaned	Other	Security Hole	Security Note	Security Warning
mail.foo.bar	general/tcp	5			1	4
mail.foo.bar	http (80/tcp)	24		14	8	2
mail.foo.bar	https (443/tcp)	30		14	8	8
mktg.foo.bar	general/icmp	1			1	
mktg.foo.bar	general/tcp	3			1	2
router.foo.bar	general/icmp	1			1	
router.foo.bar	general/tcp	3			1	2
sales.foo.bar	general/tcp	2			1	1
sales.foo.bar	general/udp	2	1			1
svr1.foo.bar	general/tcp	4			1	3
svr1.foo.bar	general/udp	2	1			1
svr1.foo.bar	http (80/tcp)	22		12	8	2
svr1.foo.bar	https (443/tcp)	28		12	8	8
svr1.foo.bar	unknown (1755/tcp)	1	1			
svr3.foo.bar	general/tcp	4	1			3
svr3.foo.bar	general/udp	2	1			1
svr3.foo.bar	http (80/tcp)	24		14	8	2

svr3.foo.bar	https (443/tcp)	30		14	8	8
svr3.foo.bar	smtp (25/tcp)	5		3		2
vpn.foo.bar	general/tcp	4	1			3
vpn.foo.bar	http (80/tcp)	23		14	7	2
vpn2.foo.bar	general/icmp	2			2	
vpn2.foo.bar	general/tcp	5			2	3
vpn2.foo.bar	general/udp	2	1			1

4.11 Query: OSList

The OSList select query produces a list of hosts and the operating systems that they are running:

```
SELECT CleanResults.[Host Name], CleanResults.InfoCleaned
FROM CleanResults
WHERE (((CleanResults.InfoCleaned) Like "Nmap found that*"));
```

Host Name	InfoCleaned
mail.foo.bar	Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98
svr1.foo.bar	Nmap found that this host is running Windows NT 4 SP3, Microsoft NT 4.0 SP5-SP6, Windows NT 4.0 SP 6a + hotfixes
svr3.foo.bar	Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98
vpn.foo.bar	Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98

4.12 Query: OSCount

The OSCount select query counts the operating systems that are present on the scanned network:

```
SELECT OSList.InfoCleaned, Count(*) AS [Count]
FROM OSList
GROUP BY OSList.InfoCleaned;
```

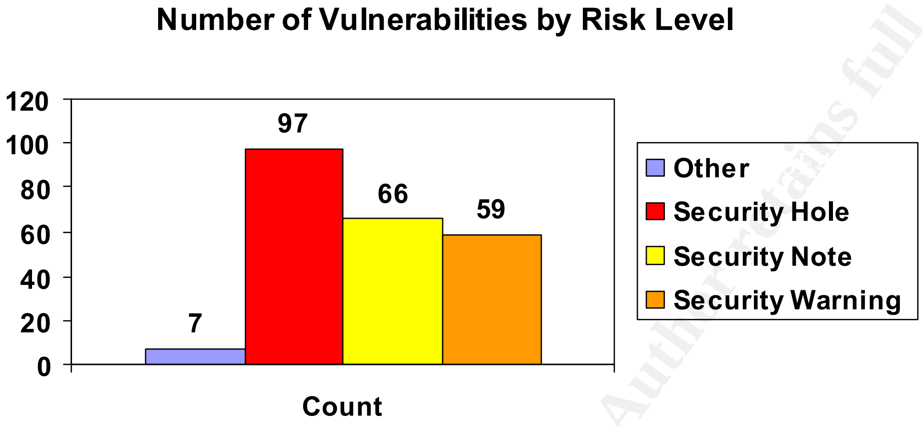
InfoCleaned	Count
Nmap found that this host is running OpenBSD 2.8 (X86), Windows NT4 / Win95 / Win98	3
Nmap found that this host is running Windows NT 4 SP3, Microsoft NT 4.0 SP5-SP6, Windows NT 4.0 SP 6a + hotfixes	1

5 Graphs and Charts

Using the above queries, it is possible to create high-level graphs and charts that illustrate the penetration test results in a meaningful way. The Microsoft Graph function is used in an Access report to create the following graphs and charts.

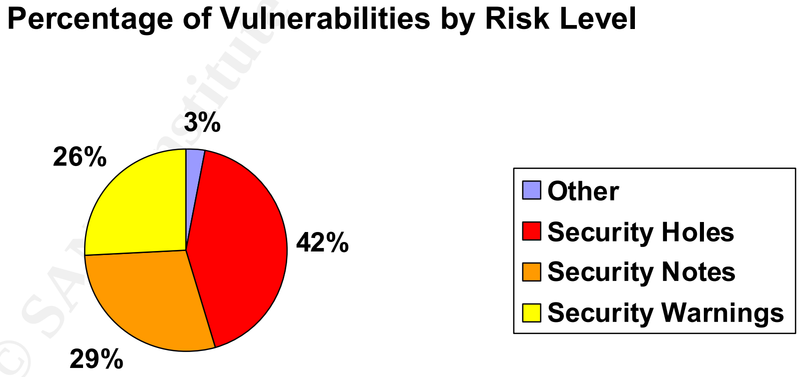
5.1 Graph: Number of Vulnerabilities by Risk Level

This bar graph is created with the CountIssues query. If placed in the summary section of a report, this graph will total issues found across all hosts. If placed in the details section, this graph will total for each host individually.



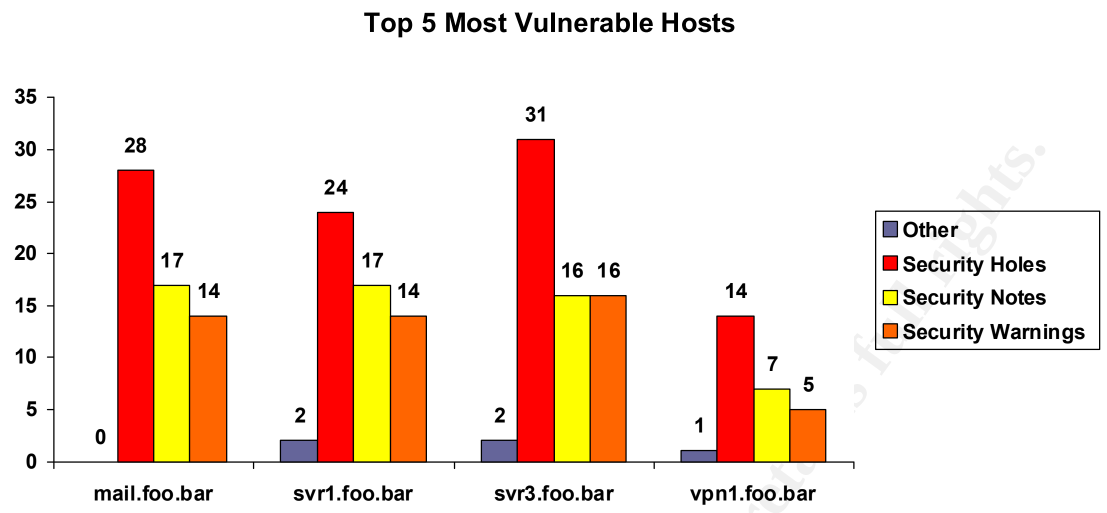
5.2 Graph: Percentage of Vulnerabilities by Risk Level

This pie graph is created with the CountIssues query. If placed in the summary section of a report, this graph will total issues found across all hosts. If placed in the details section, this graph will total for each host individually.



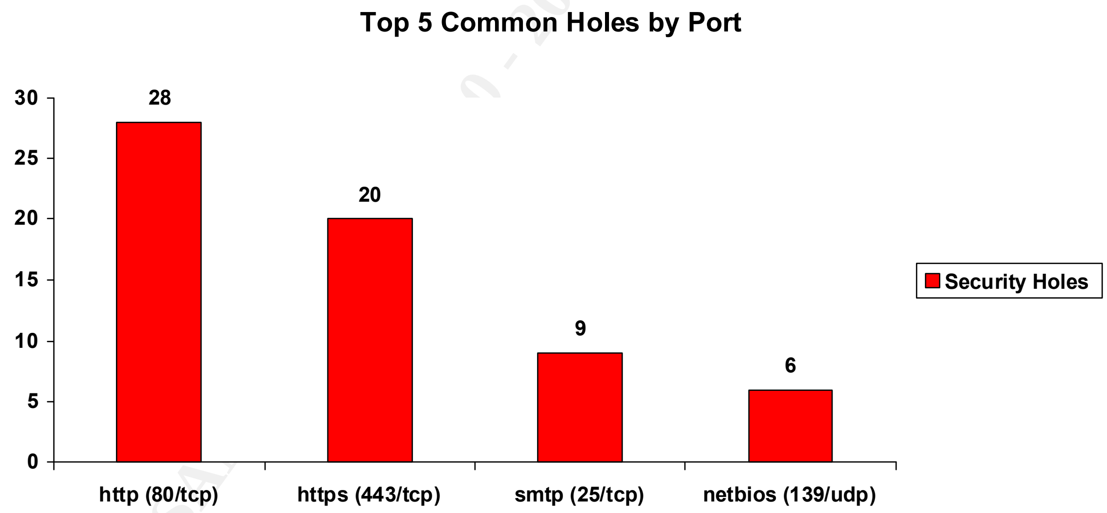
5.3 Graph: Top 5 Most Vulnerable Hosts

This bar graph is created with the Top 5 Hosts with Severity query. It presents a useful view of the most vulnerable hosts on the network for easy analysis.



5.4 *Graph: Top 5 Common Holes by Port*

This graph is made using the Top 5 Common Holes by Port query. This graph represents a total of the most dangerous services running on the network.



6 The Final Report

Using the queries and graphs developed in this paper, a single report can be made. The report can be constructed by creating a Microsoft Access report with the following sections:

- Client name, test date, auditor name. Text fields can be used as input parameters to the report. See the Microsoft Access Bible[10].
- A list of systems that were tested by Nessus (Hosts query).

- A count of systems that were tested by Nessus (HostCount query).
- An Executive Summary section containing:
  - An introductory paragraph describing penetration testing.
  - A graph counting the network vulnerabilities by risk level (CountIssues query).
  - A graph showing the percentage of network vulnerabilities by risk level (CountIssues query).
  - A graph counting the operating systems in use on the network (OSCount query).
- A Vulnerability Summary section containing:
  - A detailed graph with a breakdown of the five most vulnerable hosts (Top 5 Hosts with Severity query).
  - A graph of the top 5 common holes by port (Top 5 Common Holes by Port query).
  - A graph of security holes by category (Host Holes Totals query).
  - A table of vulnerabilities by port and risk level (Host Severity Summary query).
- A Recommendations section containing:
  - A list of general recommendations for this client. Checkbox fields can be used as input parameters to the report. See the Microsoft Access Bible[10].
  - General recommendations for all clients.
- A Host Details section containing:
  - A graph counting the host vulnerabilities by risk level (CountIssues query).
  - A graph showing the percentage of host vulnerabilities by risk level (CountIssues query).
  - Detailed penetration results for the client, sorted by port.
- A Report Footer section containing:
  - Security definitions.
  - Confidentiality information, etc.

To automate the report, a macro can be written that imports the Nessus .NBE file to the Results table. See the Microsoft Access Bible[11].

[1] The Nessus project – www.nessus.org.
[2] Chan Tuck Wai. "Conducting a Penetration on an Organization." 4 October 2001. URL: http://rr.sans.org/audit/penetration_test.php (20 July 2002).
[3] Cole, Newfield, and Millican. GSEC Security Essentials Toolkit. Indianapolis, Indiana: Que Publishing, 2002. 19-29, 136-140
[4] Cascading Style Sheets Home Page - http://www.w3.org/Style/CSS/.
[5] Microsoft XML Validation Tool - http://msdn.microsoft.com/downloads/default.asp?url=/downloads/sample.asp?url=/msdn-files/027/001/766/msdncompositedoc.xml
[6] RedHat Linux – www.redhat.com.
[7] Nmap "Network Mapper" – www.insecure.org/nmap.

[8] Prague, Cary N. and Irwin, Michael R. Microsoft Access 2000 Bible. New York, NY: Hungry Minds, Inc., 1999. 767-790.
[9] Microsoft Access Assistance Center - http://search.office.microsoft.com/assistance/producttask.aspx?p=Access.
[10] Microsoft Access Bible. 618.
[11] Microsoft Access Bible. 862.