Information Security Dashboard

By

Brian Rappach

Submitted to the Faculty of the Information Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

University of Cincinnati
College of Applied Science

May 2008

Information Security Dashboard*)*

by

Brian Rappach

Submitted to
the Faculty of the Information Technology Program
in Partial Fulfillment of the Requirements
for
the Degree of Bachelor of Science
in Information Technology

Brian Rappach Date

Mark Stockman, Faculty Advisor Date

Hazem Said, Ph.D, Department Head Date

Acknowledgements

I would like to thank everyone for helping me on the journey of completing this project. I would like to give special thanks to Michelle Melendez of Reed Elsevier Technology Services for assisting me with the design and implementation of this project. Without her guidance on what was needed and how things were constructed I would have never been able to complete the project. I would also like to thank my wife Shannon for putting up with late nights and giving me the opportunity to succeed.

Table of Contents

List of Figures

Abstract

The Information Security Dashboard is a web-based dashboard that gathers information from LexisNexis enterprise Tipping Point security management systems (SMS). As web traffic travels through the enterprise, the SMS devices and their associated sub-devices monitor the traffic, and based on packet signatures allow or block the traffic. Information about the traffic is logged into a database on the SMS device. The challenge LexisNexis faces is that there are several SMS devices in the enterprise and monitoring them all is difficult. The Information Security Dashboard solves this problem by polling each SMS device and gathering its traffic data into a master database. After all of the data is gathered, it is formatted into charts and graphs on the information security Web site. This approach allows executives and support personnel to have an updated enterprise wide view of the web traffic flow. Executives and support personnel are then able to make informed decisions about threats and problems in the enterprise. The dashboard requires little to no administrator intervention and automatically updates itself by learning new signatures and devices. The Information Security Dashboard enhances and improves the intrusion prevention system by providing a one-stop timely information portal.

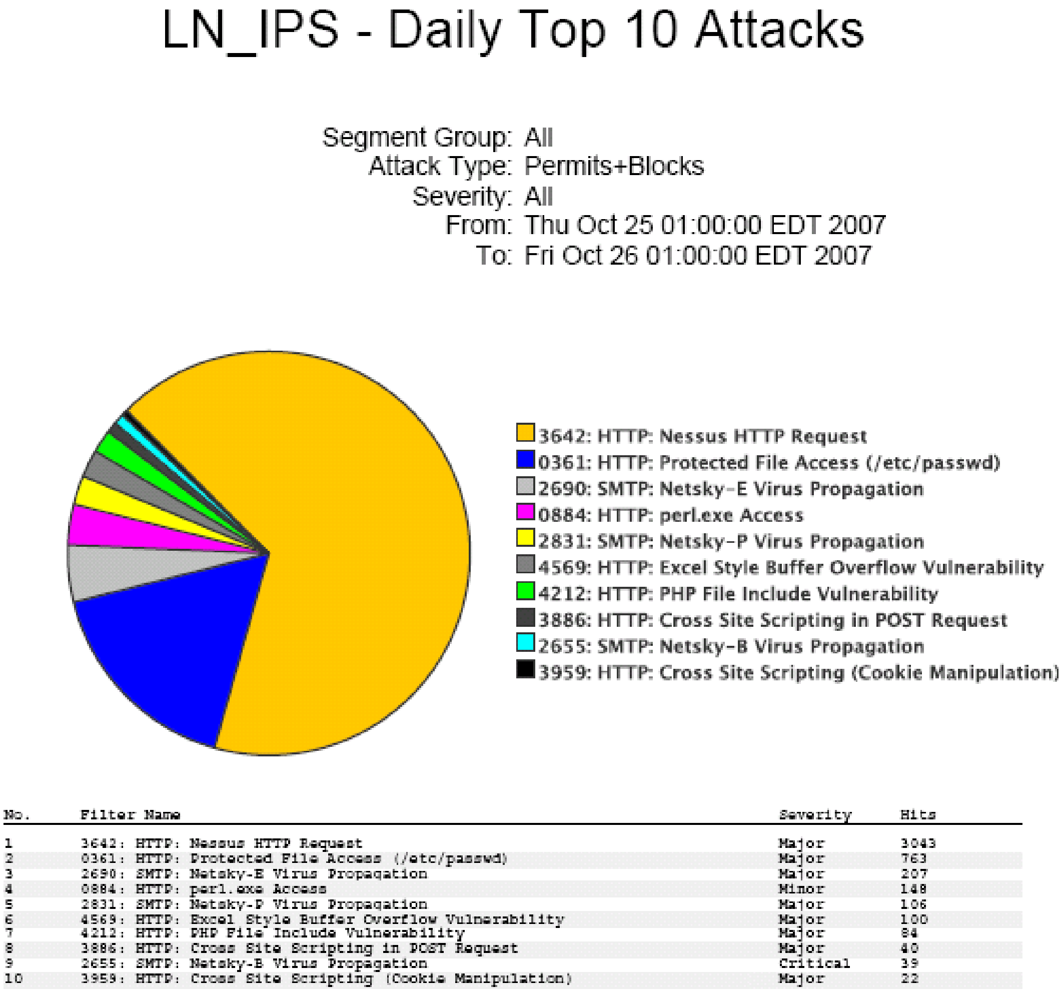
Information Security Dashboard for LexisNexis

1. Statement of the Problem

Information security professionals need to have enterprise wide reliable data to make decisions about the health and security of the enterprise. One of the best ways to monitor traffic across the network is by analyzing data that comes from the intrusion protection systems in place. These devices monitor and block traffic based on predefined filters. The devices can block traffic and notify administrators or allow the traffic while still notifying administrators.

The ability for security professionals to have a 50,000 foot view of the enterprise is helpful, but to make decisions they also need to be able to go to the granular level. This information needs to be up-to-date and accessible to individuals at any moment.

The Information Security team for LexisNexis monitors statistics and responses as part of its daily function. One of these sets of numbers is a report for the intrusion protection system (IPS) security consoles. These consoles monitor and direct traffic based on the devices they manage. The consoles provide a report that shows a pie chart and displays top ten traffic information. A sample of the report is shown in figure one.



LN_IPS - Daily Top 10 Attacks

Segment Group: All
Attack Type: Permits+Blocks
Severity: All
From: Thu Oct 25 01:00:00 EDT 2007
To: Fri Oct 26 01:00:00 EDT 2007

No.	Filter Name	Severity	Hits
1	3642: HTTP: Nessus HTTP Request	Major	3043
2	0361: HTTP: Protected File Access (/etc/passwd)	Major	763
3	2690: SMTP: Netsky-E Virus Propagation	Major	207
4	0884: HTTP: perl.exe Access	Minor	148
5	2831: SMTP: Netsky-P Virus Propagation	Major	106
6	4569: HTTP: Excel Style Buffer Overflow Vulnerability	Major	100
7	4212: HTTP: PHP File Include Vulnerability	Major	84
8	3886: HTTP: Cross Site Scripting in POST Request	Major	40
9	2655: SMTP: Netsky-B Virus Propagation	Critical	39
10	3959: HTTP: Cross Site Scripting (Cookie Manipulation)	Major	22

Figure 1: Daily IPS Report

The report does provide useful information, but the users of this information are not able to see any other information about other consoles in the enterprise (6, 3). This report only shows information from one Security Management System (SMS) device. The users would like to be able to see items enterprise wide concerning where attacks were filtered and information about the actual device that blocked the traffic. Other useful information includes which SMS it is managed by and how many of each type of attack were blocked per device. Currently, there is no mechanism providing this information. Tipping Point had offered Tipping Point pro services to come in and work with information security on getting more from the current reports, but a workable solution was unable to be completed(8).

The second portion of the problem concerns the graphing of information from surveys (8.) The users currently have to manually gather the data and build Excel graphs. The graphs are then uploaded to a Web page to be viewed by the security team to quickly glean information about route security. Manually gathering the data and turning it into graphs is a time consuming operation and again not dynamic.

Solution Description

There is a definite need to have a dynamic information security dashboard created. The dashboard will be autonomous and gather information about devices without administrator intervention. The primary focus of the dashboard is to give information security users reliable and timely information that can be updated as often as a dynamic Web page is loaded. The dashboard will also enable users to make better and faster decisions about threats.

The dashboard will communicate with the SMS consoles through the application program interface (API) provided by Tipping Point (1). In the beginning of this project, 27 devices were polled for alert information. As other SMS devices were identified in the enterprise, they were added and their sub devices polled. This number has increased to 60 devices and will go even higher when another SMS module is added in the next few weeks. The dashboard collects all of the alert and device data into a SQL database created from the original MYSQL databases stored on the SMS consoles. The report framework was created using SQL report manager services and rendered and displayed into a Share Point 2007 report viewer Web Part.

2. Design Protocols

The areas of emphasis for this project are networking, database, web design, and programming. This project is a blend of four areas of study. The networking devices involved require knowledge of IPS technology, the ability to communicate with the devices in code requires programming, and the dynamic web page requires web design knowledge. A SQL relational database holds all of the alert information.

3.1 User Profiles

The three types of users of the Information Security Dashboard are network services engineers, Information security specialists and executives. Although each user type has access to the same information, the level at which they view it is more important.

3.2 Network Services

Network services users could be any regular users of the system. They have access to the page displaying the reports. This type of user has the ability to print the different reports as well as export them to a more desirable format. This user case only differs in what level they view the information. They would be concerned with the specific devices and segments that experience alerts.

3.3 Executives

Executive users need only a high level view of the alert activity. The have access to the page displaying the reports. This type of user has the ability to print the different reports as well as export them to a more desirable format. This user case only differs from Network Services by what level they look at the information. This type of user would be concerned with what is happening at the business unit level. This user has the ability to drill down to more specific information if desired.

3.4 Information Security

Information Security users have the same rights as both of the first two use cases, but they have a special function to perform. This type of user is responsible for maintaining the code, database, reports, and SharePoint Web site. There is one Information Security user who has been chosen to perform this task.

3.5 Use Case Diagram

Almost all of the users of this system access the very same information. The use case diagram in Figure 2 below demonstrates how each type of user will interact with the dashboard. Only information security will have the ability and responsibility to maintain the dashboard.

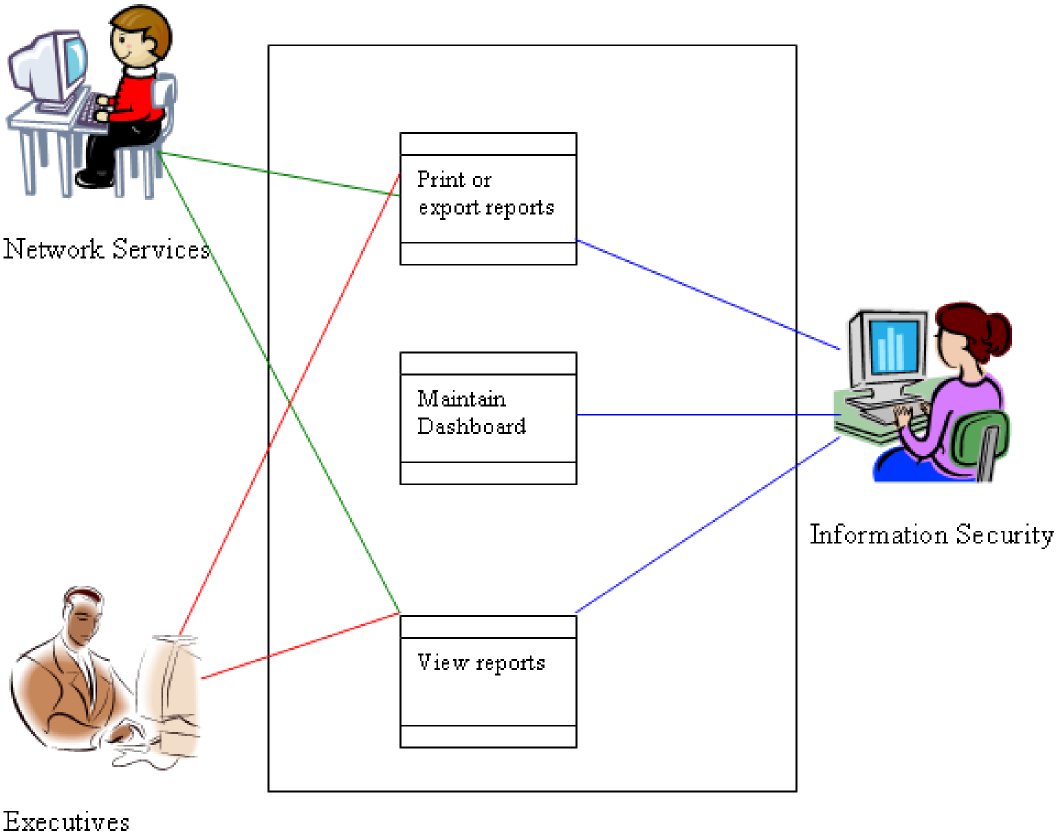


Figure 2: Use Case Diagram

3.6 Database Diagram

The database diagram below in Figure 3 details the four main tables from the database as well as how they relate to each other.

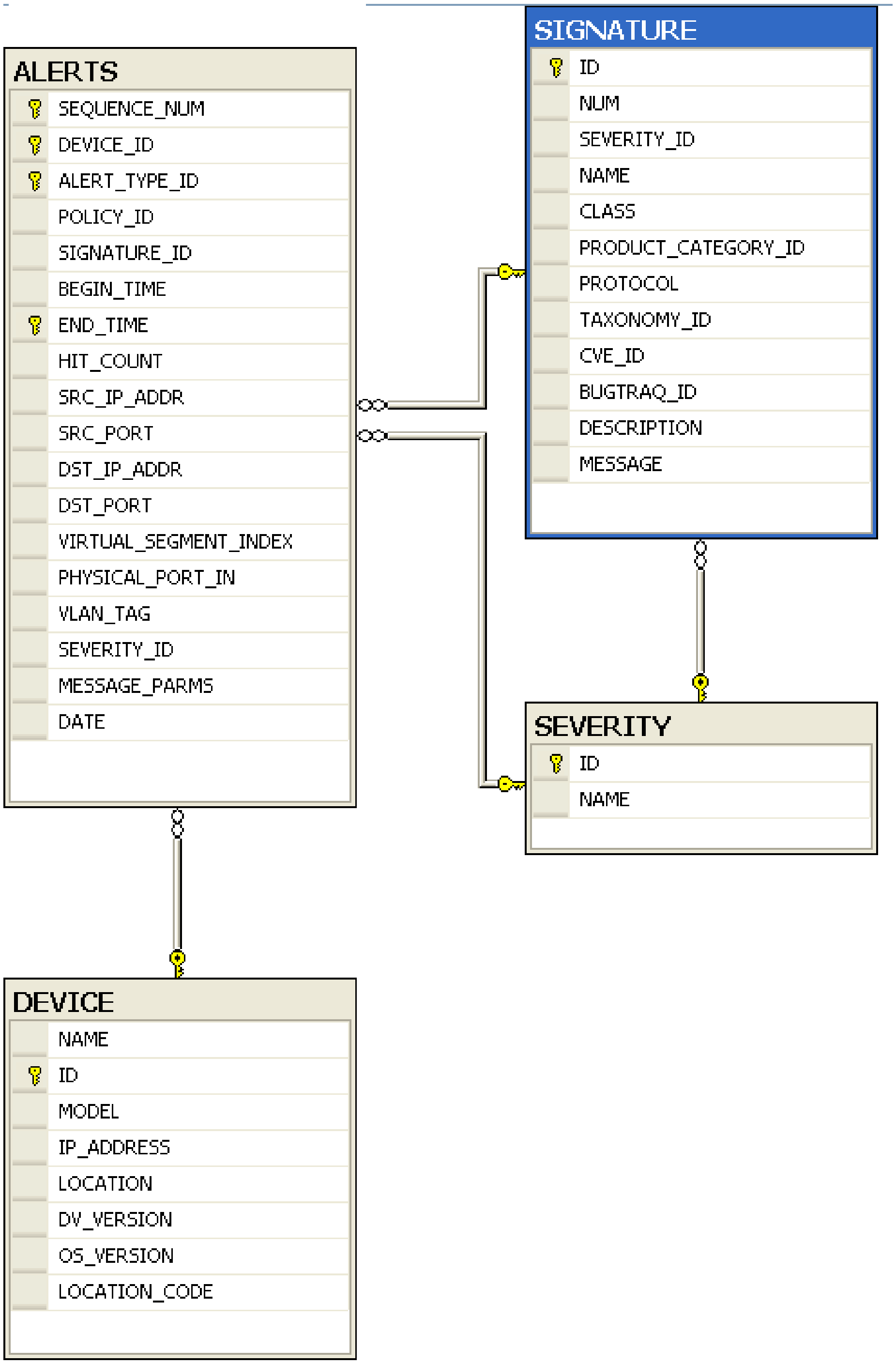
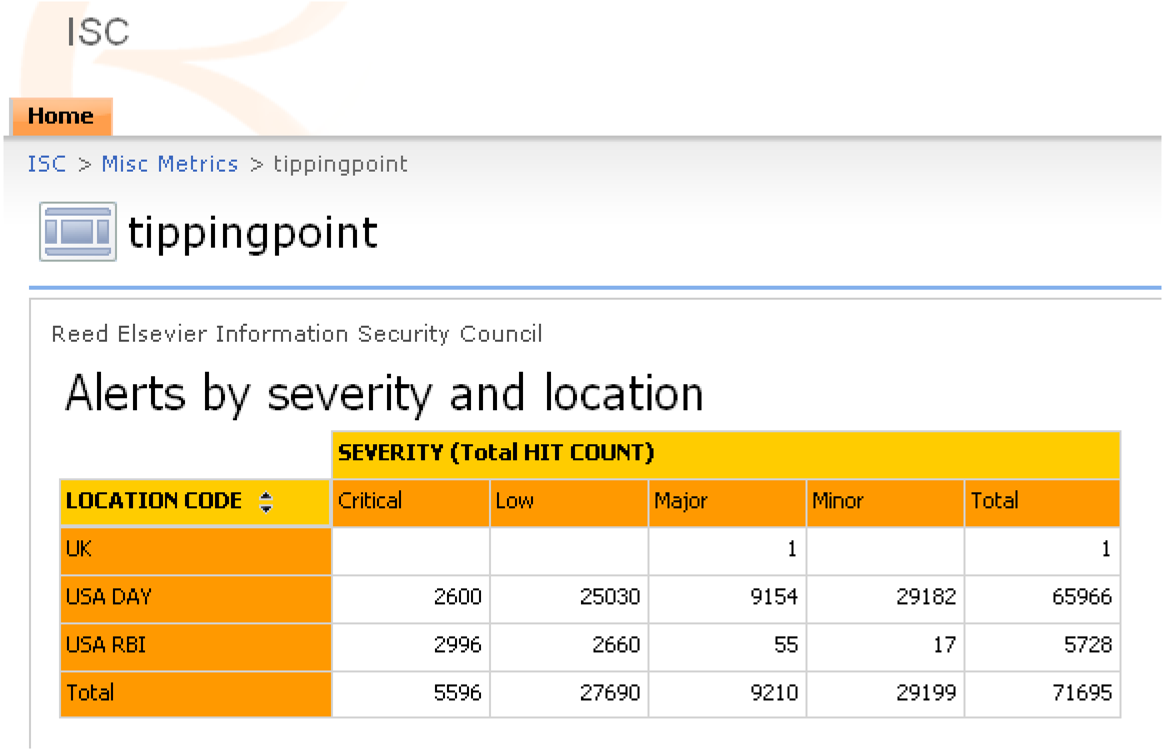


Figure 3: Database Diagram

3.7 User Interface

The user interface is comprised of three sections. The first section that users will see shows them all alerts generated today by severity and location. See Figure 4.



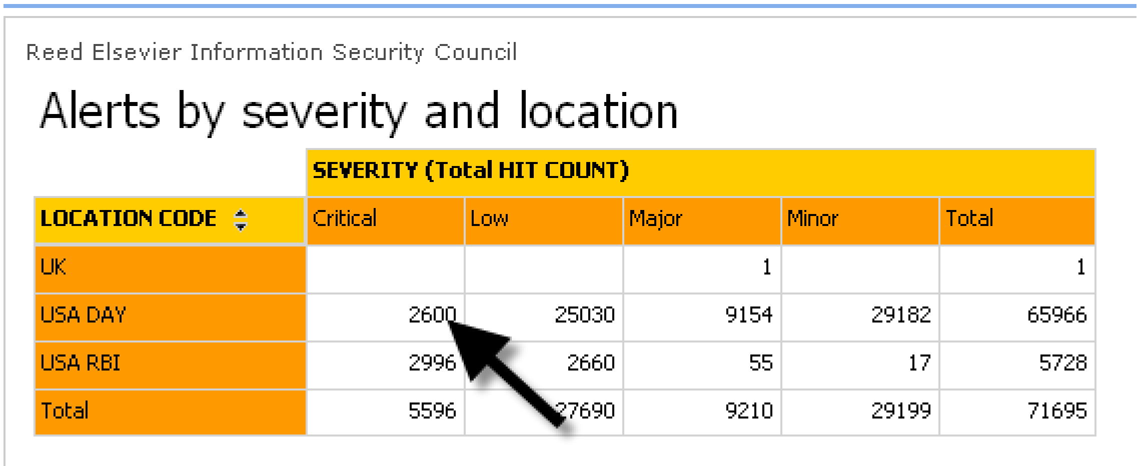
Reed Elsevier Information Security Council

Alerts by severity and location

	SEVERITY (Total HIT COUNT)				
LOCATION CODE	Critical	Low	Major	Minor	Total
UK			1		1
USA DAY	2600	25030	9154	29182	65966
USA RBI	2996	2660	55	17	5728
Total	5596	27690	9210	29199	71695

Figure 4: Alerts by Severity

By clicking on the number of hits as shown in Figure 5, a user can drill into the report and see the signatures that generated the hits.



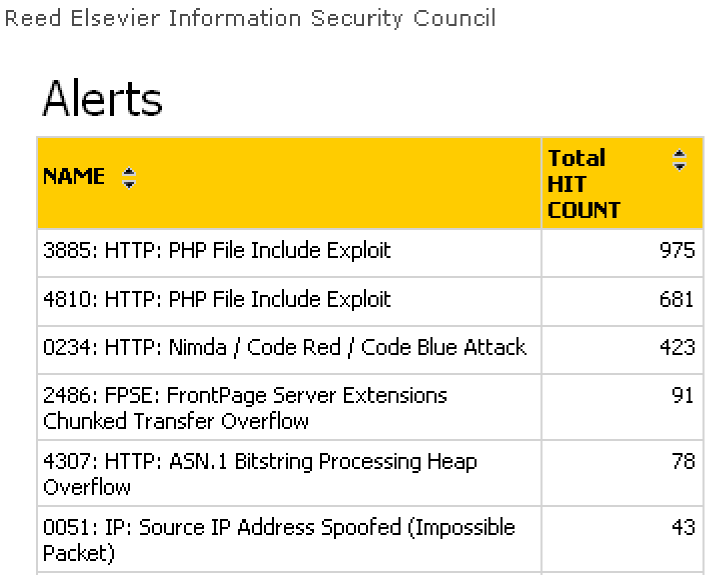
Reed Elsevier Information Security Council

Alerts by severity and location

	SEVERITY (Total HIT COUNT)				
LOCATION CODE	Critical	Low	Major	Minor	Total
UK			1		1
USA DAY	2600	25030	9154	29182	65966
USA RBI	2996	2660	55	17	5728
Total	5596	27690	9210	29199	71695

Figure 5: Click on the Number of Hits

The screen that is shown to the user lists the hits in descending order. An example of this type of report is shown in Figure 6.



Reed Elsevier Information Security Council

Alerts

NAME	Total HIT COUNT
3885: HTTP: PHP File Include Exploit	975
4810: HTTP: PHP File Include Exploit	681
0234: HTTP: Nimda / Code Red / Code Blue Attack	423
2486: FPSE: FrontPage Server Extensions Chunked Transfer Overflow	91
4307: HTTP: ASN.1 Bitstring Processing Heap Overflow	78
0051: IP: Source IP Address Spoofed (Impossible Packet)	43

Figure 6: Alert Report

The second part of the Information Security Dashboard user interface shows the user in pie chart form, the number of hits by location in Figure 7, the total number of hits enterprise wide in Figure 8 and how many of those alerts were blocks versus a warning message in Figure 9

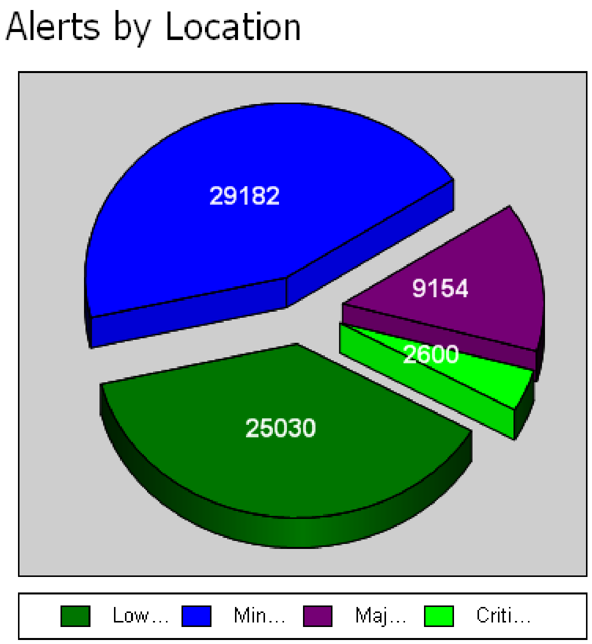


Figure 7: Alert by Location Pie Chart

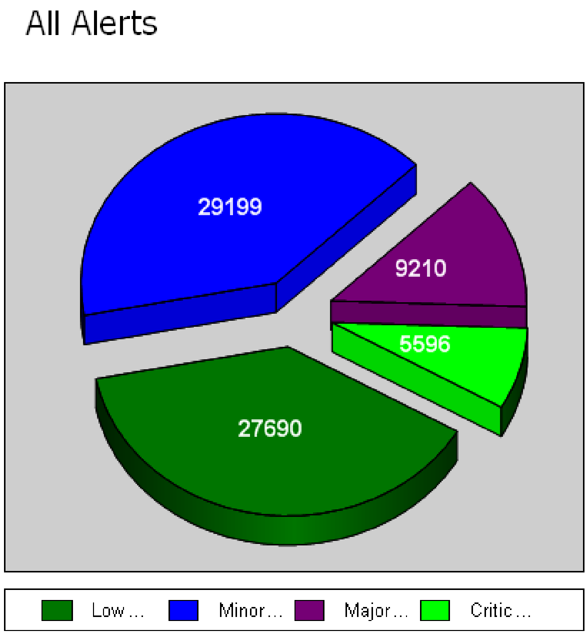


Figure 8: All Alerts Pie Chart

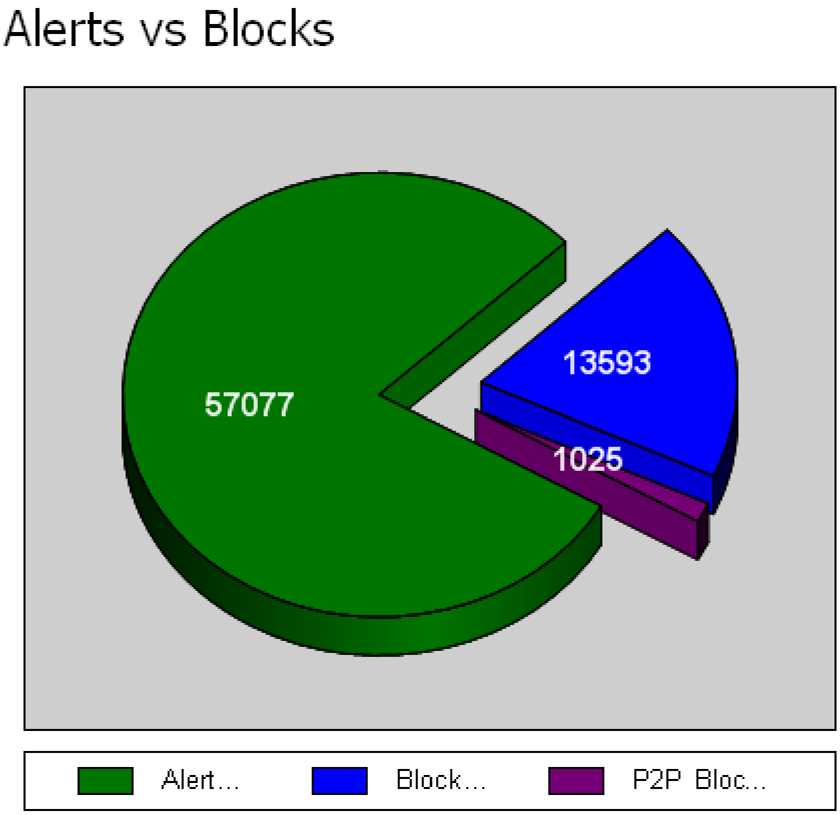
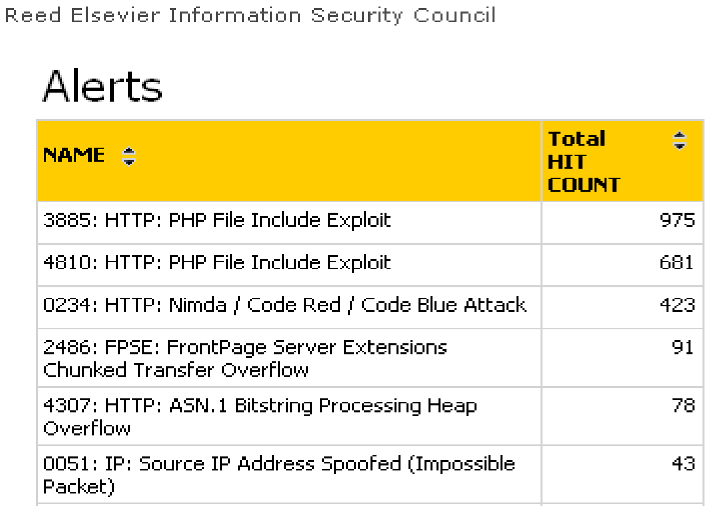


Figure 9: Alerts vs. Blocks Pie Chart

The user is again able to drill? into the pie pieces and see the signatures of that severity in descending order. (See Figure 6.)



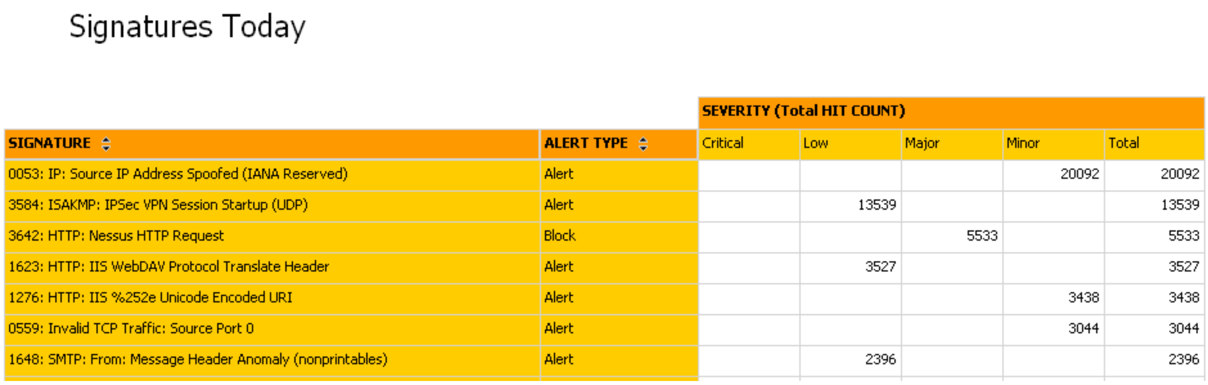
Reed Elsevier Information Security Council

Alerts

NAME ⬍	Total HIT COUNT ⬍
3885: HTTP: PHP File Include Exploit	975
4810: HTTP: PHP File Include Exploit	681
0234: HTTP: Nimda / Code Red / Code Blue Attack	423
2486: FPSE: FrontPage Server Extensions Chunked Transfer Overflow	91
4307: HTTP: ASN.1 Bitstring Processing Heap Overflow	78
0051: IP: Source IP Address Spoofed (Impossible Packet)	43

Figure 6:

The third portion of the report shows a matrix of signatures in a descending order by number of hits and severity. See excerpt in figure 10 below.

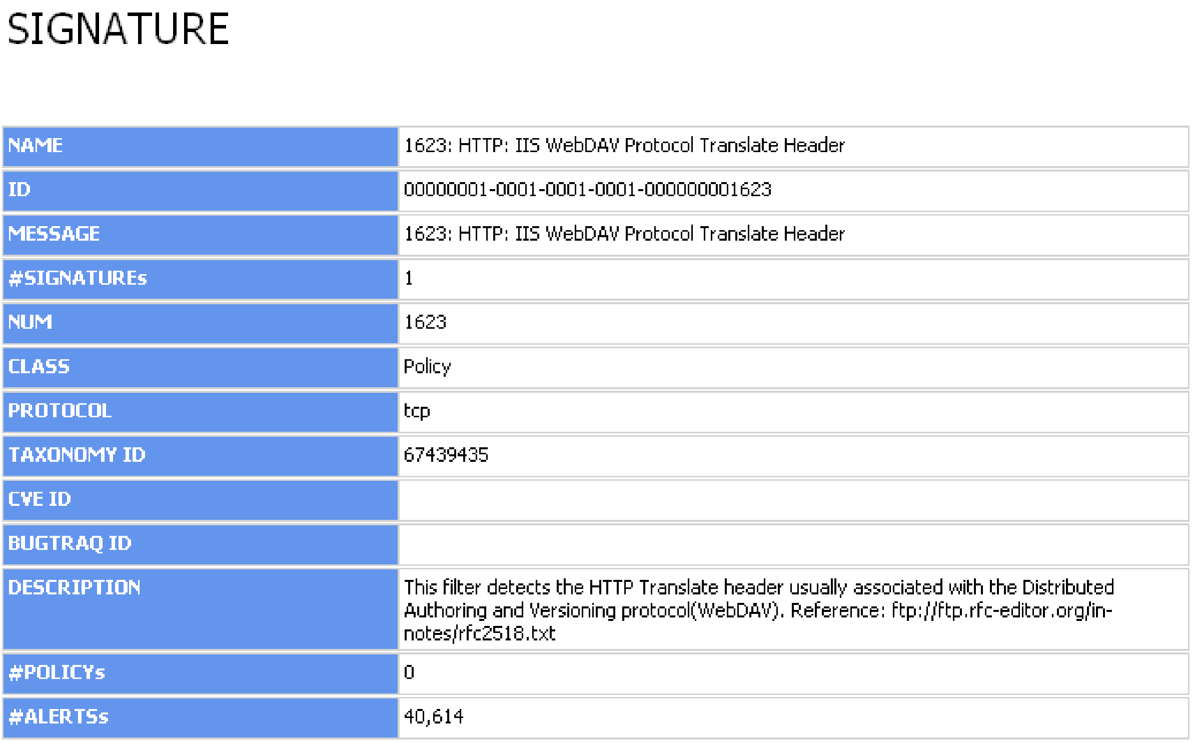


Signatures Today

		SEVERITY (Total HIT COUNT)				
SIGNATURE	ALERT TYPE	Critical	Low	Major	Minor	Total
0053: IP: Source IP Address Spoofed (IANA Reserved)	Alert				20092	20092
3584: ISAKMP: IPSec VPN Session Startup (UDP)	Alert		13539			13539
3642: HTTP: Nessus HTTP Request	Block			5533		5533
1623: HTTP: IIS WebDAV Protocol Translate Header	Alert		3527			3527
1276: HTTP: IIS %252e Unicode Encoded URI	Alert				3438	3438
0559: Invalid TCP Traffic: Source Port 0	Alert				3044	3044
1648: SMTP: From: Message Header Anomaly (nonprintables)	Alert		2396			2396

Figure 10: Signatures Today

On this portion of the report the user can drill into the signature name to find out more information about the signature as shown in figure 11.



SIGNATURE

NAME	1623: HTTP: IIS WebDAV Protocol Translate Header
ID	00000001-0001-0001-0001-000000001623
MESSAGE	1623: HTTP: IIS WebDAV Protocol Translate Header
#SIGNATUREs	1
NUM	1623
CLASS	Policy
PROTOCOL	tcp
TAXONOMY ID	67439435
CVE ID	
BUGTRAQ ID	
DESCRIPTION	This filter detects the HTTP Translate header usually associated with the Distributed Authoring and Versioning protocol(WebDAV). Reference: ftp://ftp.rfc-editor.org/in-notes/rfc2518.txt
#POLICYs	0
#ALERTSs	40,614

Figure 11: Signature Detail

3.8 Color Schemes

The color schemes were chosen for this project for two reasons. They are either Reed Elsevier Technology Services default colors or a default blue scheme. These are areas that are the different flavors of orange or they are program defaults.. The pie charts rely on the SQL reporting services default palette. A script or code snippet would have to be written to change them. The signature description screen is blue because the descriptions are informational and they are controlled and generated by the report manager itself.

4 Proof of Concept

The problems laid out for this project to solve were as follows: the current solution only provides alert data once a day for the previous day, the data is not an enterprise wide view of alerts, the current solution is also unable to be configured to show the needed data, and the current solution’s data is a flat representation which gives no information other than the numbers.

The Information Security Dashboard has a script that is run every 15 minutes as a scheduled event. This updates the database with the most current alert data. The script is also set to maintain the database once a week. It will ask each SMS console to provide a listing of devices that report to it. This list is then compared to the list of devices in the database. Any missing device information is then added to the database. The script will also ask each console for its entire list of alert signatures. This list is then also compared with the database and any new signatures are added. This solves the first problem of timely information because you are always at most 15 minutes behind.

The second issue of the alert information not being enterprise wide is also addressed by the script. The Information Security Dashboard gets alert information from four different sources. Those sources are the Dayton SMS which manages all of LexisNexis's North American IPS devices, the UK SMS which manages all of the IPS devices on the British Isles, the RBI SMS which monitors all of the IPS devices belonging to the Reed Business Information business unit, and the RE SMS that manages all IPS devices that belong to Reed Elsevier. At this point, the dashboard is getting data from almost 100 devices total all across the Reed Elsevier environment. The dashboard provides users with a truly enterprise-wide view. Because the database is a duplication and combination of all four SMS internal databases under Reed Elsevier control, information security personnel can tailor the reports to show whatever data is wanted as monitoring requirements change. This solves the issue of the original solution not being configurable.

The Information security Dashboard also solves the problem of flat data. The dashboard uses Microsoft Report Manager to create ad-hoc reports on the fly when the user loads the Web page. These are deep reports that the users can drill into to find additional information about alerts and the signatures that generate the alerts. The dashboard does this by utilizing charts, tables, and matrices report objects. These objects are linked to the database by the report server which allows related tables to be observed. The click-through reports as Microsoft calls them can be controlled to an extent that you can let the database relationships build the reports or a specific report can be built to show the exact information the administrator wants to show the user.

The Information Security Dashboard solves all of these problems while staying within the requirements of the stakeholders, support personnel and policies. The dashboard will be able to be maintained and carried forward into the future after Information Security has had training, and should remain viable well into the future. (See Appendix A.)

5 Testing

The testing plan of the Information Security Dashboard is broken up into three phases: development, certification and production. Each stage is designed to give the developer confidence that the solution does work and that it causes no harm to the environment. In the following paragraphs each phase will be detailed along with outcomes at each phase.

5.1 Development

During this phase the solution is planned and mapped out. Which technologies are available and desired are researched and quantified. Once this is complete and signed off on by the stakeholders, development is started. In the case of the information Security Dashboard, a database was created on a special development SQL server. The code was written and tested to LexisNexis support standards. The script was triggered manually by the developer at this stage. Once the dashboard was functioning as intended, Reed Elsevier Information Security Council (ISC) member Michelle Melendez was added to be able to view the dashboard. Development until she was satisfied that this was a viable solution.

5.2 Certification

During this phase, the solution will be tested by multiple users and in different situations to again assure the solution does work and that it causes no harm to the environment. The results of this phase were that the development database was reused after development and the script was placed on a web server and triggered automatically. The script logs were checked for errors and adjustments were made based on the errors seen. At this stage, users enterprise wide were added to make sure that users in different active directory containers had the same experience. The solution ran this way for almost three weeks with no major failures. The only significant log issue happened during a planned SQL server maintenance window.

5.3 Production

For the production testing phase the database will be completely rebuilt from scratch. The dashboard will be allowed to run its maintenance routine and build the database. The script will be implemented to version 1.1 with any major changes from certification being implemented. The dashboard will be allowed to run, and users will again be added to check functionality. The report server database model will be rebuilt and production reports created. The results of this phase have already been completed and the dashboard was successful in this testing phase. There were no major outages to speak of and generally all users experience the same thing. The script has been live for over 3 weeks. In March after another script update the dashboard will be shown to executives for final approval. The result of this phase was that the database was certified and is now running without intervention.

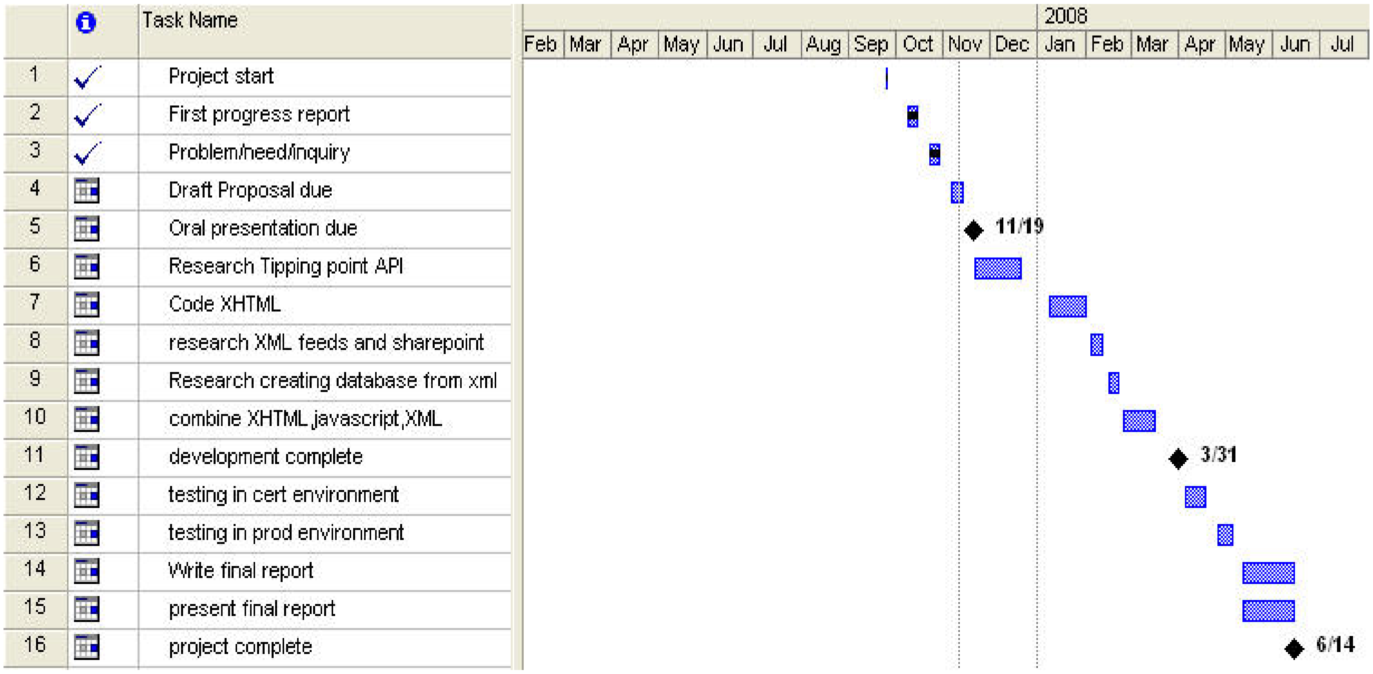
6. Deliverables

The deliverables for this project will include several forms of documentation. The first deliverable is a VISIO document model of the IPS device network. This VISIO document will allow the stakeholders to see exactly what devices are placed where on an enterprise level. The second deliverable is a database diagram of the database needed to make this project work. The IPS SMS consoles currently store everything in a MYSQL database. This database will need to be recreated in SQL to integrate properly into the environment. This document will be useful to the stakeholders if they would like to utilize it in a different manner. A data flow diagram deliverable will be generated to provide documentation on the flow of the alert information. The next deliverable will be a C# console application that polls the consoles every 15 minutes and inserts the new alerts into the database. The final two deliverables will be canned reports in SQL report manager and a SharePoint web page with report viewer web parts to display the alert information.

7. Project Planning

The time that most tasks will be completed has been estimated. These are a forecast and times to work on the project will vary based on work times and school schedule. Most of the resources will be provided by LexisNexis, but most of the development and creation of the dashboard will be done on my own time. There are several resources to consult at LexisNexis during development as well as professional contacts to assist with any programming needs that may arise. The current time track for senior design will be completing Senior Design I in Fall 2007, Senior Design II in Winter 2007, and Senior Design III in Spring 2008. Plans have been made to have the research for the API completed by January. At that time work will shift into researching the database and completing a prototype Web page by the end of March 2008. The plan is to complete this project in June 2008.

Gantt chart



	ℹ	Task Name
1	✓	Project start
2	✓	First progress report
3	✓	Problem/need/inquiry
4	▦	Draft Proposal due
5	▦	Oral presentation due
6	▦	Research Tipping point API
7	▦	Code XHTML
8	▦	research XML feeds and sharepoint
9	▦	Research creating database from xml
10	▦	combine XHTML,javascript,XML
11	▦	development complete
12	▦	testing in cert environment
13	▦	testing in prod environment
14	▦	Write final report
15	▦	present final report
16	▦	project complete

Figure 12: Gantt Chart

Technical Resources needed

This Web page will be created and supported by many pieces of hardware and software.

The tools needed include the following items.

Hardware:

- Tipping Point SMS devices
- Tipping Point IPS devices

- Web server hardware

Software:

- Windows Share Point Server
  Used to run the Web site

- Windows Share Point Designer
  Will be used to design XHTML docs and JavaScript docs

- SQL Report Server
  Will be used to manage reports

- Tipping Point SMS API
  API designed by Tipping Point to provide access to the internal

Database:

- Microsoft Visual Studio.net
  May be used to write an interface with the API

- Microsoft Windows Server 2003
  Used as an OS for the web server

- Microsoft SQL Server
  Used to create and control the database

8. Proposed Budget

As shown below in figure 13 the total retail cost of this project to start from scratch is substantial at $1,512,341.11. LexisNexis already has the infrastructure in production and available. Total cost for this Project is 0 dollars as everything needed is provided by the company.

Item	Description	Retail cost	Quantity	Actual cost
TippingPoint API	Interface provided by TippingPoint	0		0
Windows server 2003	With 5 CAL provided by LexisNexis	$707.87	1	0
SQL Server 2005	With 5 CAL provided by LexisNexis	$3,383	1	0
Visual studio 2005 pro	Provided by Lexis Nexis	$373.45	1	0
Share point designer 2007	Provided by Lexis Nexis	$199.79	1	0
Share point server	Provided by Lexis Nexis	$3,988	1	0
SQL Report Server	Provided by Lexis Nexis	0	1	0
TippingPoint 50	IPS Filtering device provided by LexisNexis	$4,995	12	0
TippingPoint 100e	IPS Filtering device provided by LexisNexis	$12,995	1	0
TippingPoint 200	IPS Filtering device provided by LexisNexis	$24,995	17	0
TippingPoint 200e	IPS Filtering device provided by LexisNexis	$14,995	3	0
TippingPoint 400	IPS Filtering device provided by LexisNexis	$34,995	15	0
TippingPoint 600e	IPS Filtering device provided by LexisNexis	$34,995	6	0
TippingPoint 1200	IPS Filtering device provided by LexisNexis	$64,995	3	0
TippingPoint SMS	IPS management console provided by LexisNexis	$9,995	3	0
	Total cost	$1,512,341.11		**0**

Figure 13: Budget (2, 3, 7, 9, 10)

Conclusion

The Information Security Dashboard needs to be implemented. The dashboard will enable LexisNexis information security team members to see information that is up-to-date and enterprise-wide. This dashboard was not an easy objective to complete as it will drew on four different technologies: networking, database, Web design, and programming. I feel confident that LexisNexis will find value in this project. Several groups within the company such as Information Security, Network Support, and management will benefit as a result of this effort. A better understanding of the threats received on an enterprise level can greatly assist network administrators to better understand which filters are working and which are not. The dashboard will be able to be carried forward in the future by other teams and groups to improve its capabilities.

References

1. 3Com Corporation, "TippingPoint SMS External Interface Guide version 2.5.1" 2002-2007

2. 3Com Corporation, www.3com.com/other/pdfs/solutions/en_US/ny_pricelist.pdf 600e 400, 11-11-07.

3. Chordia, Panna. Network services, LexisNexis. Personal interview and e-mail 10-26-2007.

4. Cronin, Leo. CISO, LexisNexis. Personal interview 10-16-2007.

5. Huffman, Mat. Database instructor. Personal interview 10-23-2007.

6. "Intrusion prevention IPS", http://www.tippingpoint.com/ , 2007.

7. KCDA purchasing Cooperative, www.kcda.org/pricerecaps/Technology/3Com_KCDA_Price_List_072905.xls, 11-11-07.

8. Melendez, Michelle. Information Security, LexisNexis, Personal interview, 10-26-2007.

9. NH&A anti-virus, firewalls, and security solutions, http://nha-fl.com/tpflpricing.htm, 11-11-07.

10. Pricegrabber.com, www.pricegrabber.com , 11-11-07.

11. Ricard, Jonathan. Desktop Support Manager, LexisNexis. Personal interview at monthly one on ones 2007.

12. Romer, Calvin. Network services, LexisNexis. Personal interview 10-26-2007.

13. Stockman, Mark. Networking advisor. Personal interview advisor meeting 10-16-2007.

14. Wager, Jim. Desktop Support Lead, LexisNexis. Personal interview 9-17-2007.

Appendix A

Below you will find the turnover document that was generated for information Security.

Portions have been redacted to keep information confidential

Reed Elsevier Technology Services

Reed Elsevier

TippingPoint Dashboard

Abstract: This document describes the requirements and management of the TippingPoint Dashboard. This Document will give the reader an understanding of the underlying structure of the controlling script and report server making this dashboard possible.

Document Number:	
Document Version	0.2
Release Date:	January 31, 2008
Document Author:	Brian Rappach
Document Owner:	Information Security

Contacts

Any questions or comments about this document may be directed to:

Brian Rappach 937-865-6800 x54066 brian.rappach@reedelsevier.com

Document Record

People involved in the preparation of this document:

Function	Name
Author	Brian Rappach

Review list

Reviewed by	Date

Change History

Version	Date	Revision Description	Changed by
0.1	Jan 31 08	Initial Draft	Brian Rappach
0.2	Feb 12 08	Added more information to sql database recovery	Brian Rappach

Table of Contents

Preface

The purpose of this document is to detail the Information Security Dashboard for the Information Security team members and executives desktops within the Reed Elsevier Enterprise. The use of the Dashboard is restricted to only individuals who have been granted access to the REISC report server.

Additionally, this document will include a comprehensive look at the controlling script so that the dashboard will remain a dynamic work in process project that can be updated as usage requirements change. The Report Manager process will be covered so that any person reading this document can make changes, recreate reports after a data loss or create new reports if the person is given permission. The final part of the document will cover how the reports integrate into Reed Elsevier Enterprise SharePoint environment.

Introduction and Dashboard Overview

The Dashboard has four major components. The first component is a C# console script that is triggered every 15 minutes on an ETS Dayton web services server. The Second portion of the dashboard is a collection of reports that reside on an ETS Dayton Report Manager server. The Third Major portion is a SharePoint Web Page Maintained by the Information Security Council. The final part is the SQL database that houses all of the information about the devices and alerts.

These four pieces together combine to create a dynamic web page that provides enterprise reporting of IPS filter activity. Viewers are able to see what signatures have been triggered and where they were triggered in the last 24 hours. This approach is more agile than the current system of an e-mail report that gets distributed every morning.

The C# Console Application

The C# console applications job is to maintain the database. This includes the addition of new signatures, devices, and alerts to the database. The script runs every 15 minutes and maintains its own error log. This error log will report and general errors as well as any SQL errors with additions to the database.

The Main Method

The main method of the script is the controlling function of the entire script. This is the starting and ending point for the script. Figure A below is a screen shot of the main method.

REDACTED

Figure A

The first thing the Script does is disable Certificate errors that may be generated. This SSL_DisableCertificateErrors routine should not have to be changed unless SSL certificate errors are going to be tracked. The second call getepoch calls a function that calculates the number of milliseconds since the UNIX epoch or 1-1-1970 at 8 am. The full function can be seen in figure B.

REDACTED

Figure B

The script does this because TippingPoint uses the number of milliseconds since epoch to calculate when alerts were generated. The function takes the current date and subtracts the epoch date. This time span is then converted to milliseconds and rounded to the whole millisecond. The timestamp variable is stored globally and can be used anywhere in the script as the time that the script started to run. The next line creates a variable called prevtimestamp and calls a function called getstamp that will inspect the registry of the server running the script and look for the last time the script ran.

Once the getstamp routine is finished it returns back to the main method and onto the next command. The next routine setstamp sets a registry value so that the time the script was run can be "remembered". The next command starts the maintenance routine called maintenance. The maintenance routine pictured in figure C first checks the registry to see if the counter is above a predetermined threshold. Initially this is set to the equivalent number of times it will run in one week.

REDACTED

Figure C

As you can see from the figure the script first places itself in device maintenance and then sets a variable to the connection string of a monitored SMS device. The variable usasmsip and uksmsip contain the actual IP addresses of the consoles. It also set a global variable to add a location to each device that is returned from the ComApi method. The ComApi method is a common routine used by the maintenance function and the main method to insert items into the database.

The first part of the ComApi method sets up and establishes a connection with the SQL Database so that devices and alerts can be added. Figure D shows this portion of the routine.

REDACTED

Figure D

After this part of the code is run the connection to the database is open and ready to use. The first line of the SMS consoles response is also loaded into the line variable.

Figure E shows the next part of the ComApi routine that takes the steam and before doing anything to it checks to see if any of the maintenance switches are turned on. If the device maintenance switch is on the script will add the location to the stream before it is sent to the database.

REDACTED

Figure E

If the device maintenance switch is not on it checks to see if the signature maintenance switch is on. If it is it passes the stream directly through to the database. If the signature maintenance switch is not on it checks to see if the device maintenance switch is off. The logic here is that it checks to see if the first 2 switches are on and the third time it checks to make sure device maintenance is off. This prevents a signature or device stream from being updated incorrectly. So all general alerts go through the third process of having the alert time converted from milliseconds to human readable time. This was done to make it easier to sort the alerts later in the SQL database.

Add an SMS to be monitored

To add a new SMS device to be monitored first gather the new IP address. Create a new variable to hold the IP and assign the IP to it. See figure F

REDACTED

Figure F

Now add a line to the main method to include this devices request along with a call to the ComApi. See figure G.

REDACTED

Figure G

Be sure to add the device in the maintenance section so all sub IPS devices are added to the db. Be sure to call the ComApi and set the location code as well.

REDACTED

This should be all that is needed to update the script. When TippingPoint releases newer versions of the API these calls may need modified.

The SQL Report Manager Reports

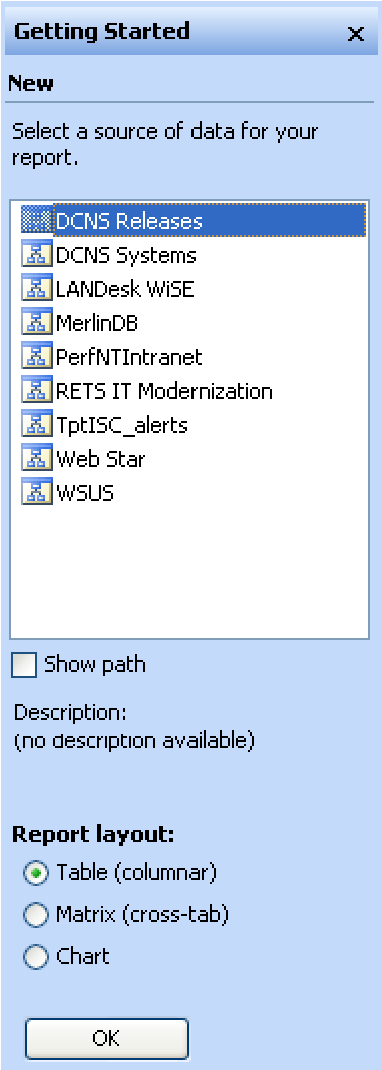
The SQL Report Manager potion of this solution is housed in the ETS Dayton Report Server located at **REDACTED** in the REISC folder. The folder is owned by Leo Cronin and Michelle Melendez.

REDACTED

The datasources folder contains a model of the SQL database which allows the reports to be generated. To be able to create or view reports you need to be a member of an active directory group. To view reports you need to have your id added to the **REDACTED**
group. To be able to create reports you need to have your id added to the **REDACTED** group.

Creating or recreating a report

To create or recreate a report navigate to the report server with a browser (**REDACTED**). Next double click on the REISC folder. Select the report builder button. **REDACTED** This will start report builder and you will be presented with a list of models to use on the right hand side.



Getting Started

New

Select a source of data for your report.

- DCNS Releases
- DCNS Systems
- LANDesk WiSE
- MerlinDB
- PerfNTIntranet
- RETS IT Modernization
- TptISC_alerts
- Web Star
- WSUS

□ Show path

Description:
(no description available)

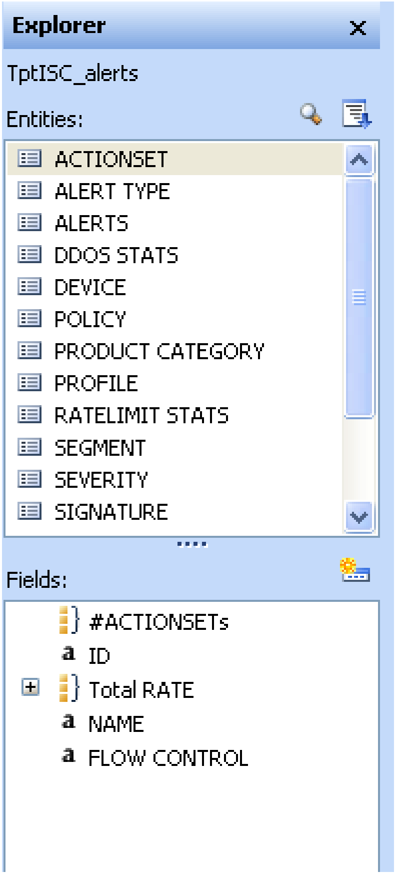
Report layout:

● Table (columnar)

○ Matrix (cross-tab)

○ Chart

Select the TptISC_alerts model and choose your report style of chart, matrix, or table. On the left hand side of the window you will see all the tables in the database.



Explorer

TptISC_alerts

Entities:

- ACTIONSET
- ALERT TYPE
- ALERTS
- DDOS STATS
- DEVICE
- POLICY
- PRODUCT CATEGORY
- PROFILE
- RATELIMIT STATS
- SEGMENT
- SEVERITY
- SIGNATURE

Fields:

- #ACTIONSETs
- ID
- Total RATE
- NAME
- FLOW CONTROL

Using these entities you will be able to create tables, matrices', and charts. For more information on how to style the reports there are multiple sources on the web. Start your search at www.microsoft.com and branch out from there.

Rebuilding the database model

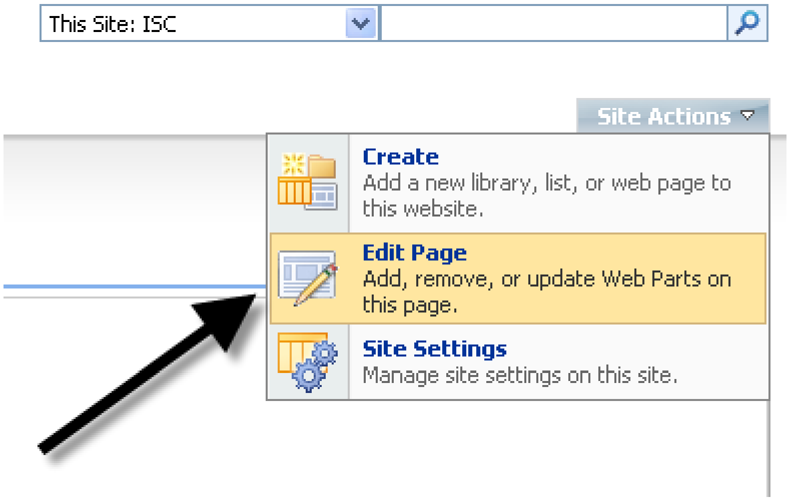
To rebuild the database model double click on the datasources folder and find the database object. There will be two types of files in here. The two files are the model and the data source itself. To recreate the model open the data source itself. There should be a button near the bottom called create model. Select the button and overwrite the current model. Do not change the name as then all of your reports will need to be changed to associate with this new model. This will only work if the model has not been modified to a manual configuration. This section of the help file should be changed if it does to reflect the manual process.

The SharePoint Web Site

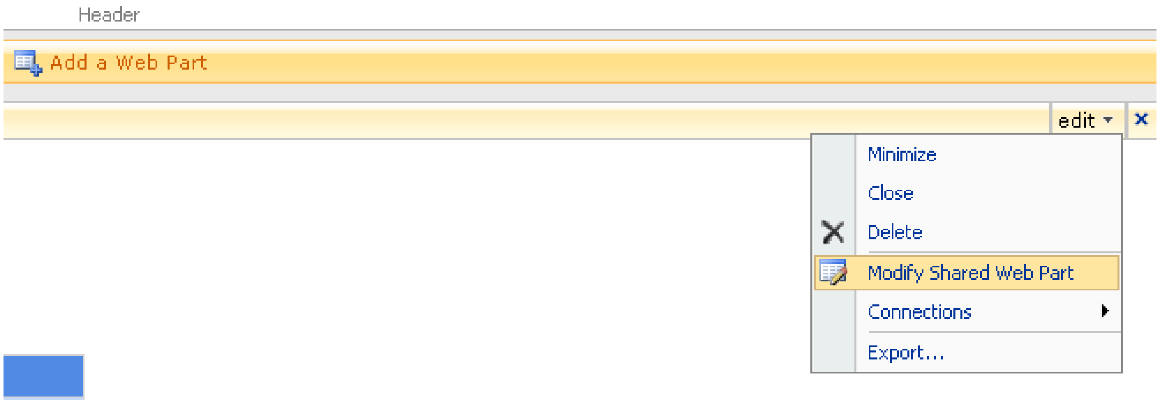
The SharePoint Web Site is housed on the ISC home page. The page can be found at **REDACTED** and is owned by Michelle Melendez. The page called TippingPoint houses all of the current reports.

Modifying the SharePoint Web Parts

Make sure you have been given access to the ISC page as an editor by the owner. Navigate to the Tippingpoint page identified above in section 5. Click on site actions in the upper right hand section of the screen. A drop down will come out and you should select edit page.


This Site: ISC
Site Actions
Create
Add a new library, list, or web page to this website.
Edit Page
Add, remove, or update Web Parts on this page.
Site Settings
Manage site settings on this site.

In the upper right hand corner of a Web Part click edit and then select modify shared Web Part



This will open the modification windows where you must specify the report server **REDACTED**. You must include the leading / when entering this.

REDACTED

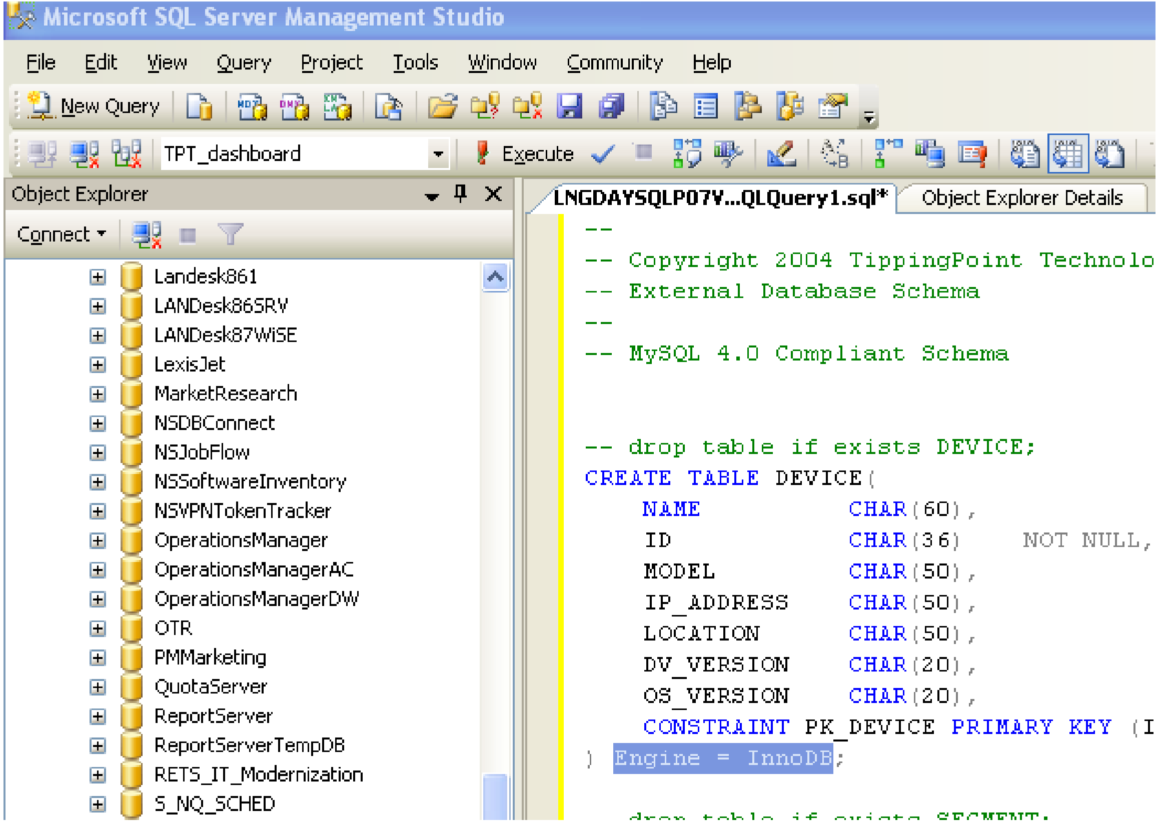
Make any other additions to the appearance and layout and select ok. The report should now populate the Web Part.

The SQL Database

The SQL database that is used to run this application is a conversion of the TippingPoint MYSQL databases that the SMS's run on. The current environment does not make it practical to use MYSQL. The conversion was done using the data dictionary and schema function of the TippingPoint API. Further modifications were made to the database to better serve the needs of the script.

Recovering from complete database destruction

First a blank SQL database will need to be created and the API's output of the schema call applied. Caution the output of the Schema will be in MYSQL format. You will need to remove the innodb calls after each table. Dump the entire output to a new query as seen in figure O.

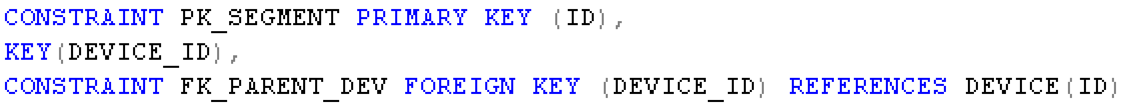


```
--
-- Copyright 2004 TippingPoint Technolo
-- External Database Schema
--
-- MySQL 4.0 Compliant Schema


-- drop table if exists DEVICE;
CREATE TABLE DEVICE(
    NAME            CHAR(60),
    ID              CHAR(36)    NOT NULL,
    MODEL           CHAR(50),
    IP_ADDRESS      CHAR(50),
    LOCATION        CHAR(50),
    DV_VERSION      CHAR(20),
    OS_VERSION      CHAR(20),
    CONSTRAINT PK_DEVICE PRIMARY KEY (I
) Engine = InnoDB;

-- drop table if exists SEGMENT;
```

Figure O

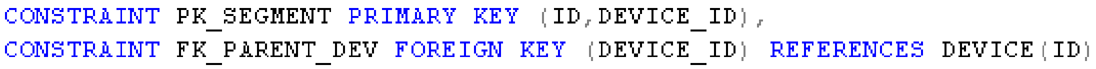
Start with the device table and remove the Engine = InnoDB as this is a MYSQL function. Highlight this first table and execute the query. This table should create normaly.

The next table is the segment table you will again have to remove the innodb statement but you will need to make some adjustments to the constraints. Because the primay key and foreign keys have multiple fields as the key you will need to format this for SQL. To do this change the constraint statements from



```
CONSTRAINT PK_SEGMENT PRIMARY KEY (ID),
KEY(DEVICE_ID),
CONSTRAINT FK_PARENT_DEV FOREIGN KEY (DEVICE_ID) REFERENCES DEVICE(ID)
```

To



```
CONSTRAINT PK_SEGMENT PRIMARY KEY (ID,DEVICE_ID),
CONSTRAINT FK_PARENT_DEV FOREIGN KEY (DEVICE_ID) REFERENCES DEVICE(ID)
```

This will allow for normal key creation. Follow the same pattern for each table removing the innodb and changing the constraints if there are multiple field keys. Also to note SQL will not accept unsigned fields. If this error occurs just remove the word unsigned from the offending line. Also of interest is you can not specify a size on a smallint field. If sql throws this error remove the (#) from the column. Remove the two FK constraints for the threshold stats table. They are not found in the referenced table and should not be needed. Go through each table adding it to the database by changing the query so that each create returns with no errors.

Once this is complete then the data dictionary API call can be used to repopulate all of the tables. Once all of the tables are rebuilt before any alerts can be added you must load certain table data first. The alerts table depends on other static tables. Use a call to the get data method of the API for the alerts table to get some alerts to try to add. When you try to add the alerts SQL will advise what static table it needs loaded before it will accept alerts. Use a call to the get data method of the API for the specified table and load it. Try to add alerts again and again SQL will notify you of the required tables.

Once you are able to add alerts with no errors you are ready to make modifications. Add a field to the device table called LOCATION_CODE. Make this a text field with reasonable length. Next add a field to the alerts table called DATE. Make this field a datetime field and you should be set to go.

Foreseen issues with the database

TippingPoint appears to make modifications to the database and API on a yearly basis. Field names are changed, added or removed. You will need to study these additions and changes carefully to make sure that it will not affect the script, reports, or database. Changes can be made directly to the script or database but for the report see the section on updating the database model for report server.

Appendix B

Terms used in this turnover document

API	Application program interface
ETS	Enterprise technology support
IPS	Intrusion Prevention System
REISC	Reed Elsevier Information Security Council
SMS	Security management system
SSL	Secure Socket Layer

Resources used for the turnover document

Microsoft Reporting	www.microsoft.com
Microsoft SQL	www.microsoft.com
Share Point	www.microsoft.com
TippingPoint	see TippingPoint API reference manual
Visual Studio	www.microsoft.com