ISYS 520 - Spread Sheet Automation

Instructor: Gove Allen, PhD

Final Project
Vulnerability Report Analyzer

By

Antonio Uriarte

Section 1

December 8, 2011

Table of Contents

Executive Summary

Security specialists utilize a vast array of tools to perform different scans on companies' systems. These tools are utilized to detect potential vulnerabilities that relate to the configuration and patch level of a given server. Upon performing a vulnerability scan, security specialist analyze the data in order to separate false positive alerts from real system weaknesses. This analysis process can be tedious and challenging due to the large amount of data produced by the scanning tools. A factor that largely contributes to the difficulty to analyze data is that not all tools generate reports that can be easily exported to a spread sheet. One of such tools is Tenable's Nessus Scanner.

The Nessus scanner is widely used by security professionals to analyze the vulnerability level of a given company. However, Nessus's reports are only made available on an HTML format. While the reports have greatly improved since the initial launch of Nessus, it is still difficult to summarize data to present it to management in a meaningful way.

The Vulnerability Report Analyzer is a program that imports Nessus' scan results and creates a report table where management can easily sort through each finding. The program can generate a summary of the top ten servers with the most critical vulnerabilities as well as a comparative graph that indicates the type of vulnerabilities per server. Users are able to create multiple reports within the spreadsheet. The program is flexible to allow users to append results of scans performed at a later date to original reports. This feature is useful to generate a single monthly scan report while weekly scans are performed. The program automatically generates pivot tables for each report in order to assist management in filtering different field. Graphs are also generated from pivot tables that can be filtered by hosts and vulnerability risk levels. This tool is particularly useful for environments that host hundreds of systems and where multiple vulnerability scans are performed.

Given than Nessus has several HTML report outputs, it is important to mention that this program currently works only with the "Detailed HTML Report (by finding)". Further functionality could include:

- Addition of other Nessus report outputs;
- Import of .csv files from other scanning tools such as McAfee's Foundstone; and
- Trend analysis to detect those servers that continually present the same vulnerabilities.

Overall, the program is a useful tool that can be utilized to gain meaningful information on the status of systems. As far as implementation of VBA concepts, the program makes use of virtually every topic discussed during the semester, from ranges, to forms and from arrays to pivot tables. I am confident that you will find this tool very useful in your analysis of vulnerability reports.

Implementation

As a security professional I know that working with vulnerability scanning tools can be very interesting and overwhelming. These tools generate large outputs that are difficult to organize, read, and analyze. Given that one of the most popular tools in the industry is Tenable's Nessus, I decided to address the problem I continually faced while performing Nessus' scans: making sense of the large data presented in HTML in a way that allows both, security professionals and management to understand the real security posture of a company's system and take the corresponding mitigating actions.

Importing Data

The first task to accomplish with the program was to import data into Excel that can be analyzed and presented to management. The latest version of Nessus, (4.4.1) uses a web-based client interface to schedule scans and obtain reports. However, Nessus only exports report data in three formats: HTML, RTF, and Nessus' proprietary format to be read by other Nessus scan (See Figure 1). It is, however, possible to obtain a .csv file on the most expensive version of Nessus; however, most business and security professionals in the industry use the basic paid version.

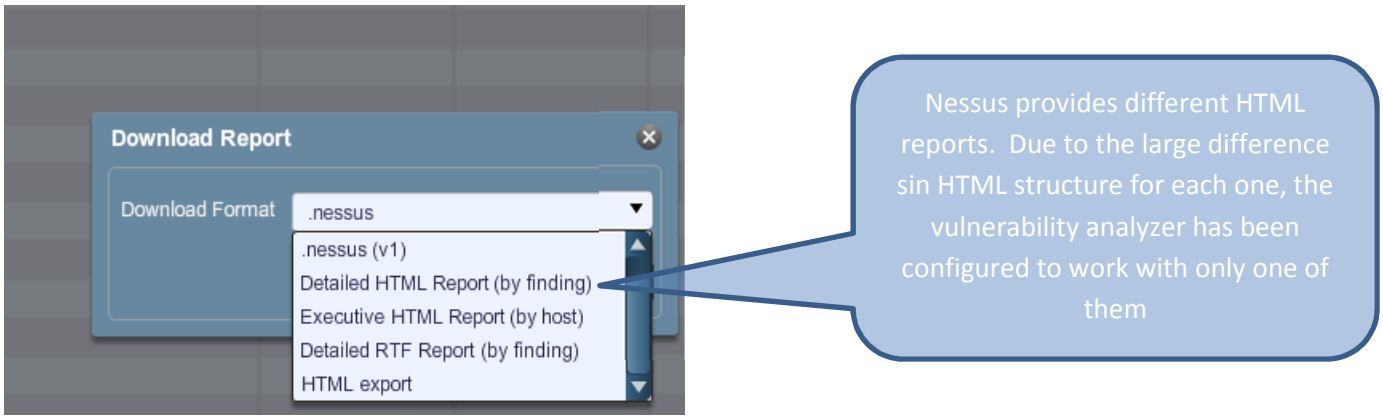


Figure 1 - Nessus Report Formats

To determine which format would be more manageable for exporting into Excel, I ran several scans against my virtual machines and exported the results in the three formats supported by Nessus. I quickly realized that using the HTML format would be the best option, particularly because of Dr. Allen's Agent class to parse through HTML.

Nessus has three different HTML report views: by finding, by host, and the plain HTML export. After studying each report and their corresponding source code I determined that the report that was most accessible to be used with the Agent class was the HTML Detail Report (by finding).

In order to make the program as robust as possible, two types of pre-configured scan policies were used: Internal Network Scan, and PCI DSS Audits. The reason for selecting those reports was that they are the most commonly used in the industry for scanning internal servers. Policies that were not tested

were External Network Scans and Web App Test. One of the main reasons for not testing those policies was because they are meant to be conducing outside a company's network. Given that I do not currently have the capability of hosting an externally facing site, I cannot perform these scans in a controlled environment.

As it was already mentioned, the program has the capability of importing the HTML files for analysis. In order to perform the import and analysis of data, a new tab called "Vulnerability Reports" was added to the ribbon (See Figure 2).

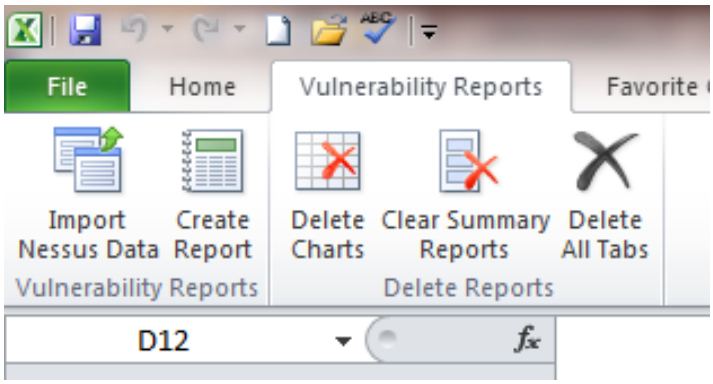


Figure 2 - Vulnerability Reports Tab

From this tab users can:

- Import Nessus HTML files to the workbook;
- Create a pivot table that summarizes the number of vulnerabilities per host and inserts a list of the top ten offending hosts in a Summary sheet;
- Delete all charts created;
- Clear the top ten offending hosts from the Summary sheet; and
- Delete all tabs containing report data.

When a user clicks on the Import Nessus Data, a form appears asking the user whether to create a new spreadsheet (where the data will be collected), or append data to an existing spreadsheet (See Figure 3). The option of appending data to an existing spreadsheet is particularly useful in the event that multiple scans need to be consolidated into one. A typical example of this situation is when a large organization performs weekly scans of different servers and wants all results to be contained in a monthly report. Currently, Nessus cannot merge scan results, this forces users to have separate reports for each scan performed.

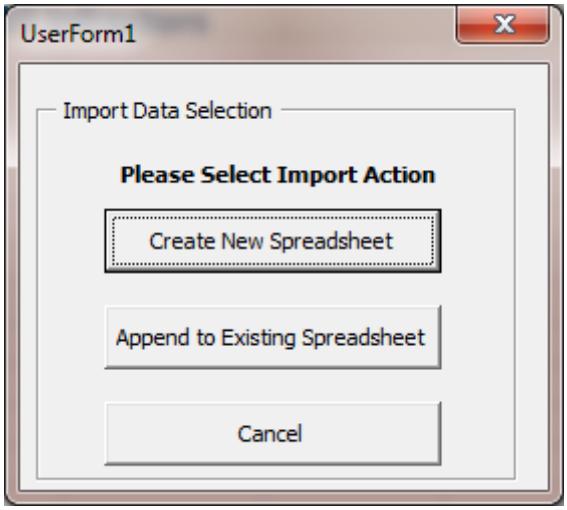


Figure 3 - Import Form

When the user clicks on "Create New Spreadsheet" an input box is presented where the user is asked to name the new spreadsheet. Naming of each sheet will depend on the naming convention for each user. For demonstration purposes, we will assume that we are generating monthly reports and that each spreadsheet will contain the information for all scans performed within a month (See Figure 4).

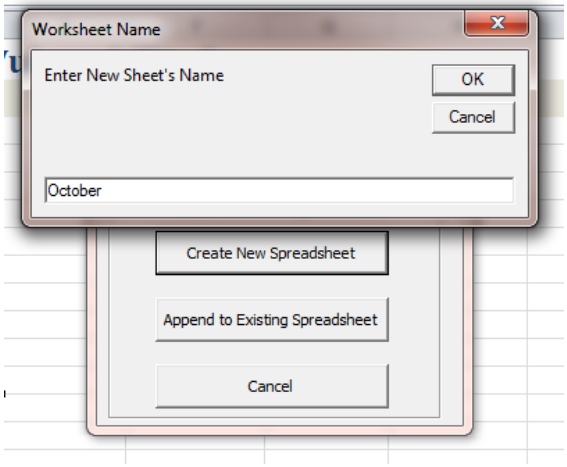


Figure 4 - Naming of Spreadsheet

After naming the sheet, the user is prompted whether he or she would like to import data at the moment. The reason behind this prompt is because if several scans are being performed for a particular time period, users can decide to import existing scans now, or just create a placeholder and add the results in the future.

If a user decides to continue with the import process, the user is presented with a file browser to navigate to the location where the Nessus files are located. In Figure 5 we can see that there are three available files. For demonstration purposes I named the files with different month names; however, all

of the scans for each file were performed on the same day. I have included the same files with my program to facilitate its evaluation.

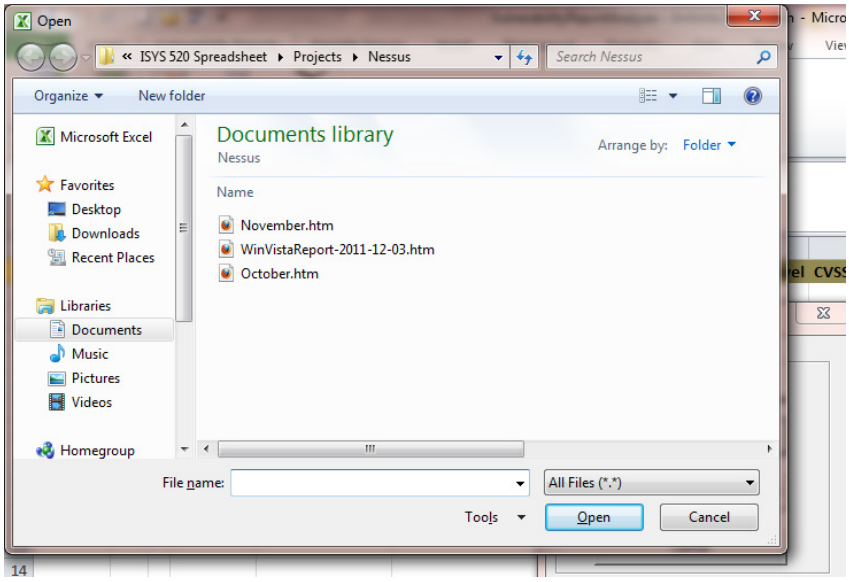


Figure 5- File Selection

Upon selecting one file (in this example the October.html was selected), the file is imported to the newly generated tab. The file is automatically formatted with headers, color lines, and auto filter to facilitate the examination of data (See Figure 6).

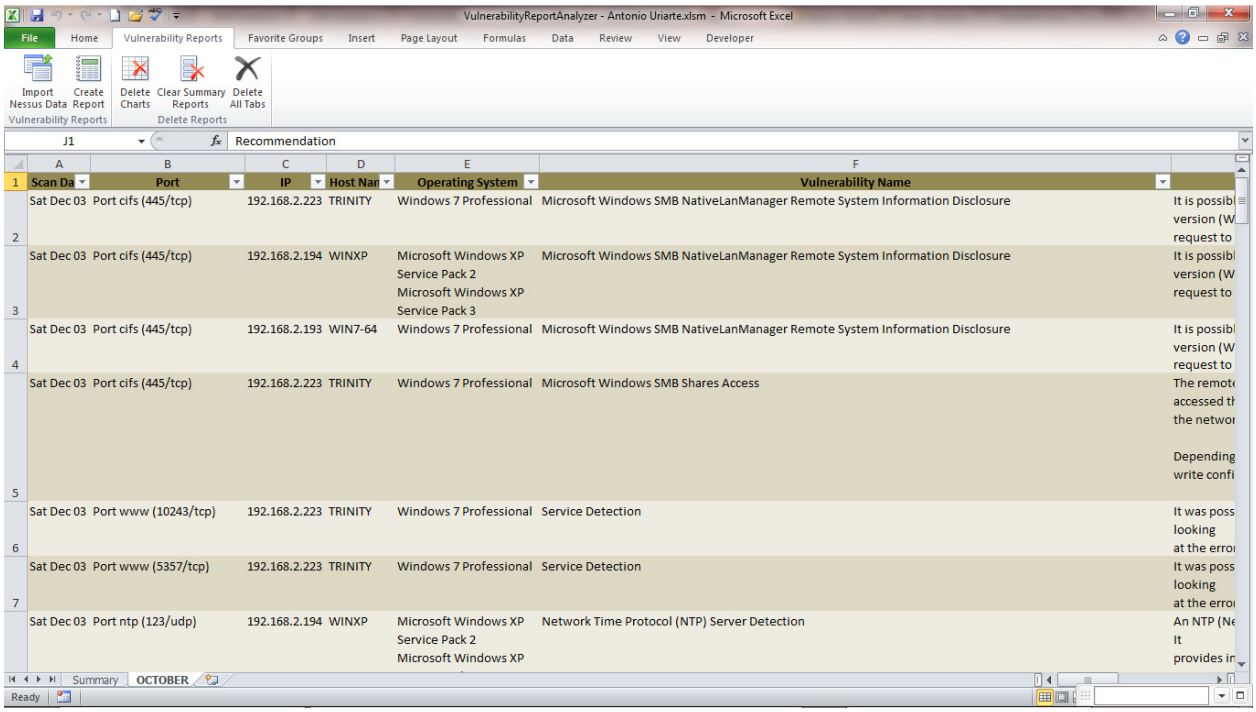


Figure 6 - Imported Data

The imported data includes the following information:

- Scan Date;
- Port scanned;
- IP address (of the scanned machine);
- Host name (named of the scanned machine);
- Operating System;
- Vulnerability Name;
- Vulnerability Description;
- Risk Level (this level is assigned by Nessus based on the common vulnerability scoring system (CVSS));
- CVSS Base Score; and
- Recommendations for vulnerability mitigation

At this point the program has completed the import of data. Users have the option of either a) import additional data to the same sheet or to a different sheet, or b) create a report that will create a Pivot table, identify the top ten vulnerable machines, and create a graph on the summary table.

To demonstrate the capability of appending data to an existing spreadsheet, we will use the scan result of a single Vista machine. To append data, the user:

- Clicks on the Import Nessus Data button on the Vulnerability Reports tab
- Selects the "Append to Existing Spreadsheet" button
- A new message box appears to determine whether the user wants to add data to the active sheet or to another existing sheet (See Figure 7)

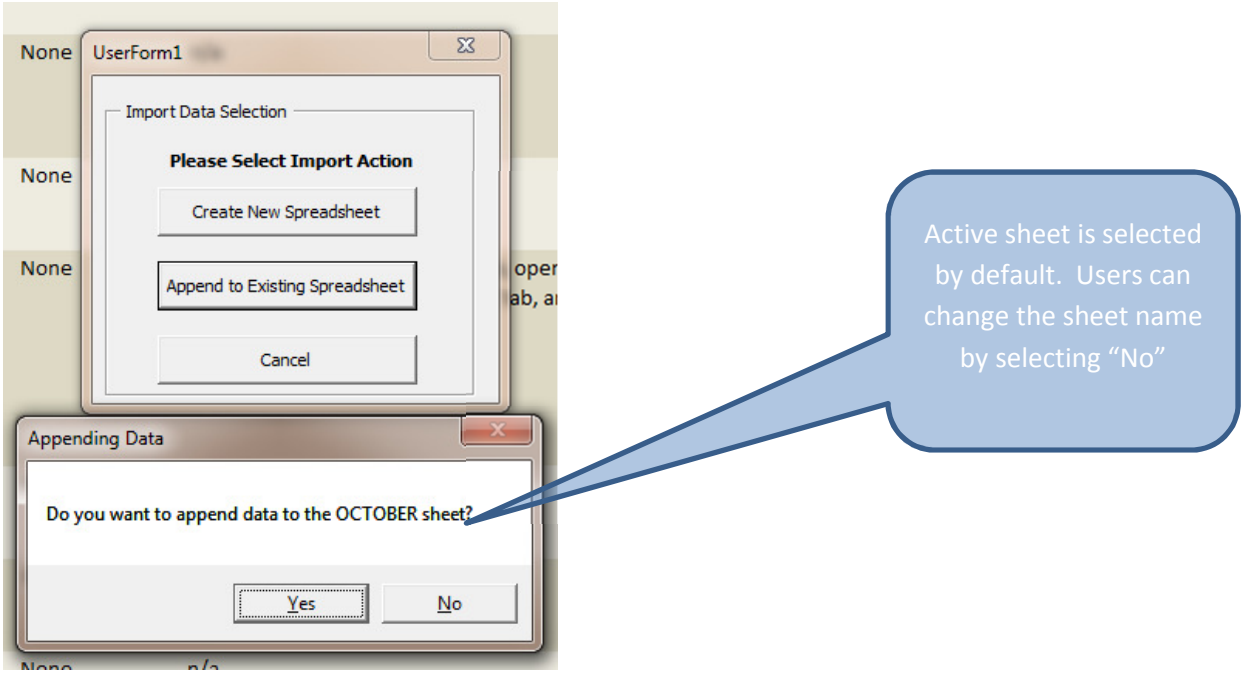


Figure 7 - Append to Active Sheet Question

- If a user selects “No”, a new dialog box appears to allow the user to enter the name of the spreadsheet to which the data will be appended. If the spreadsheet does not exist, or the name is incorrect, the user is given a warning indicating to check spelling (See Figure 8). After that, the program reverts to the first form to allow the user to restart the process.

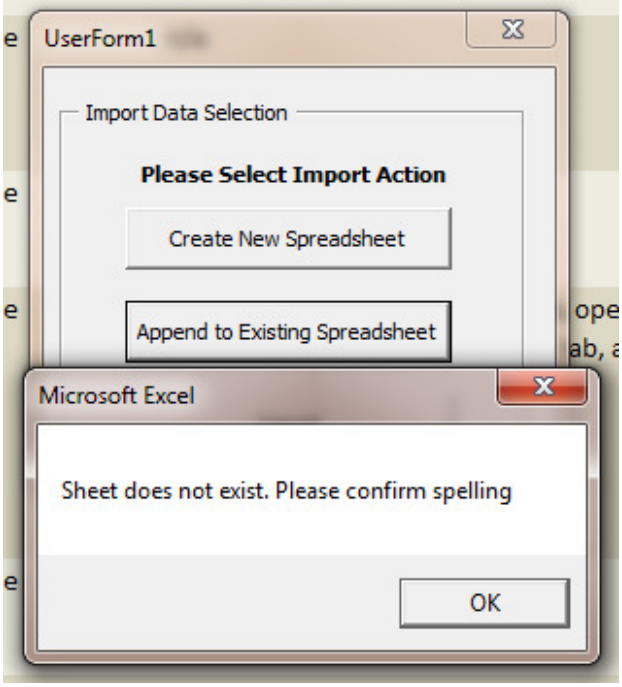


Figure 8 - Notification of Non-existing Sheet

- Once a valid sheet name has been provide, the program displays the file browser window to allow the user to select the HTML file to append (similar to Figure 5 above).
- Upon selection of the file (we selected WinVistaReport) the program imports the data into the existing spreadsheet by appending it at the bottom of the last row.

Report Generation

The Vulnerability Report Analyzer contains a front Summary sheet (see Figure 9). On this sheet the program will insert a summary of the top ten hosts with vulnerabilities. Additionally, a graph is created form a Pivot Table to represent vulnerabilities per host.

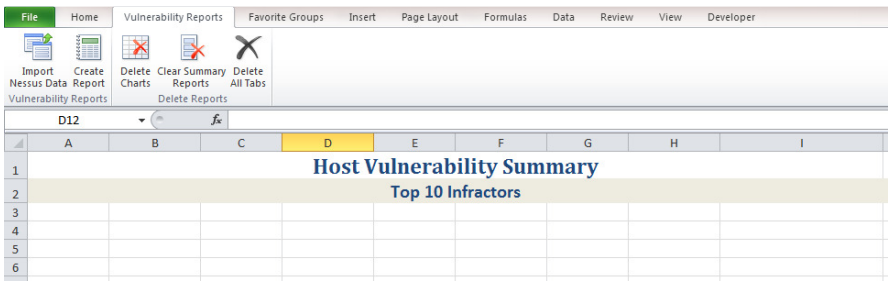


Figure 9 - Initial Summary Sheet

To generate a report, users must:

- Click on the Create Reports button in the Vulnerability Reports tab. This will bring a new form where the user can select either a) create a report from the active sheet, or b) name the sheet from which the report will be generated. As a security measure, if the user were to select to create a report from the Summary sheet, the system will create a message indicating that such action is not allowed (See Figures 10 and 11)

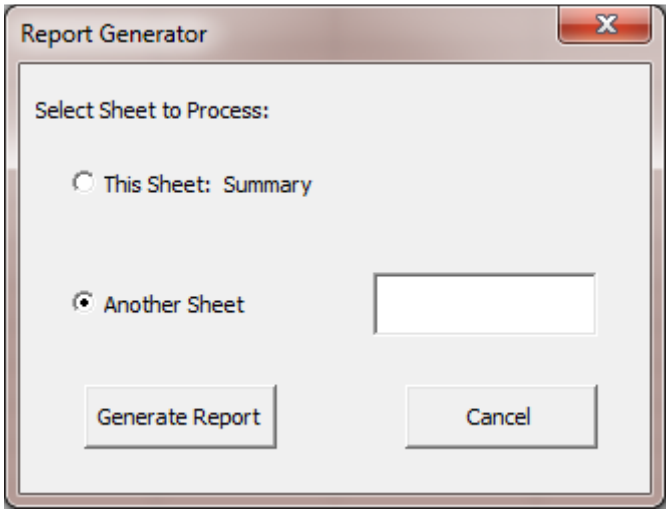


Figure 10 - Selection of Sheet for Report

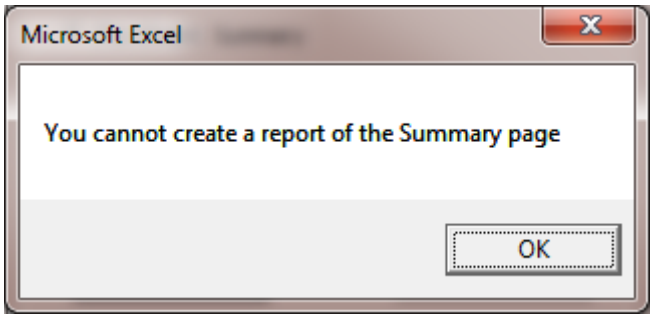


Figure 11 - Message display upon attempting to create a report of the Summary sheet

- Enter the name of a valid sheet. For demonstration purposes, we have selected the October sheet. This action automatically generates a new sheet named "Table" + <name of data source sheet>. The program automatically generates a Pivot Table on the new sheet. This table shows the number of vulnerabilities per host as well as the risk level associated to each vulnerability. The table is automatically sorted based on the most critical vulnerability (See Figure 12 below). The logic behind creating a Pivot Table with this default view is based on my professional experience when reporting to management. If management is interested in seeing additional information, the Pivot Table provides great flexibility to modify data.

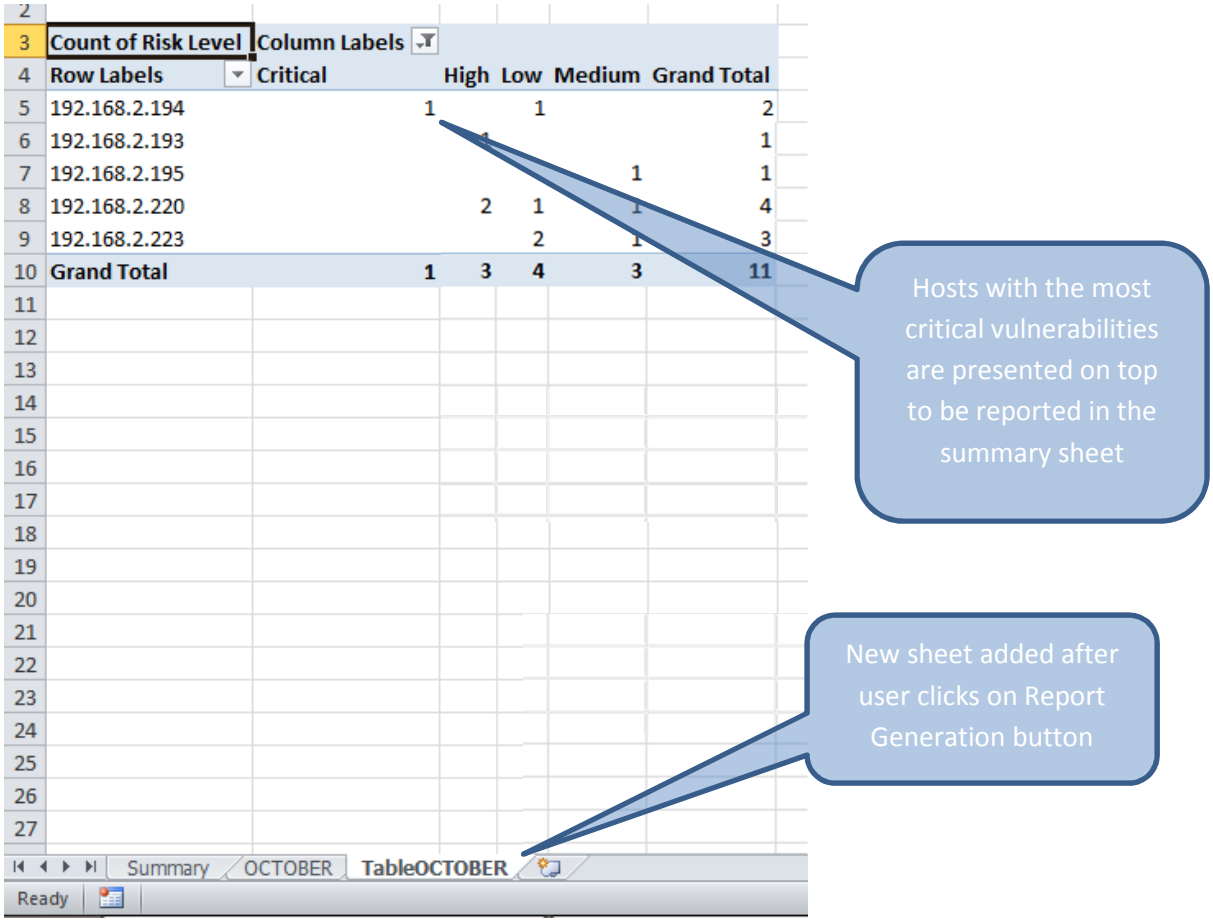


Figure 12 - Pivot Table with hosts sorted by number of vulnerabilities

Given that the table has already sorted each host based on the number of vulnerabilities, the selection of the top ten servers is simply made by choosing the first ten servers. This is done automatically by the program. Note: given that reduced number of hosts used (five) this cannot be evidenced in this document. The program automatically populates the Summary sheet with a list of the top ten servers and a total number of vulnerabilities. An automatic pivot table graph is also automatically generated. The chart is filtered by default to show only Low, Medium, High, and Critical vulnerabilities. However, the user can easily modify the filter to include vulnerabilities with no risk value (See Figures 13 and 14 below).

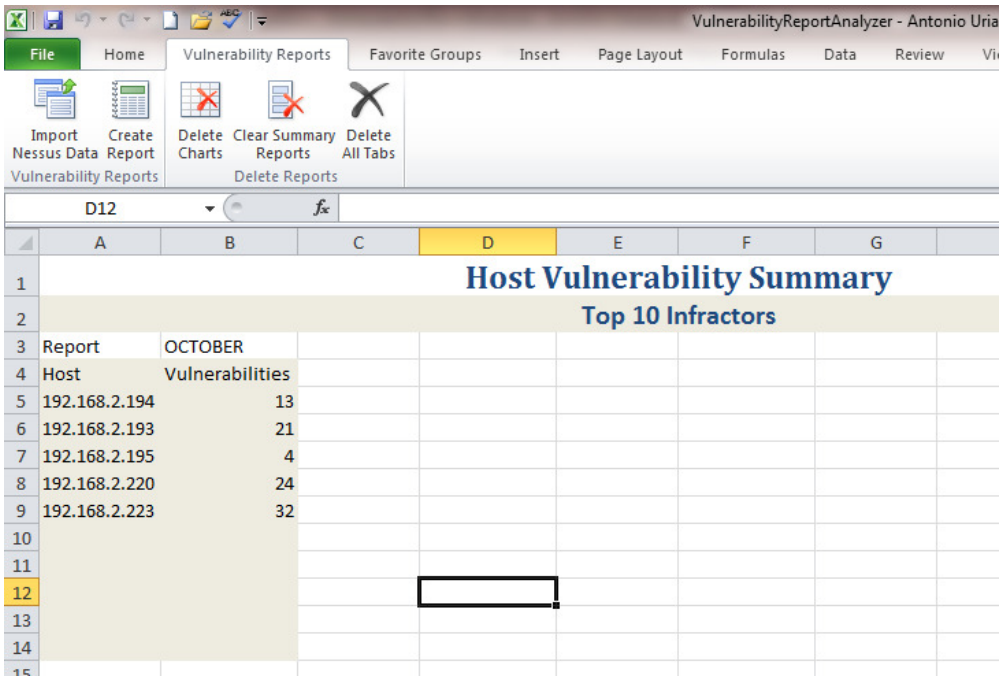
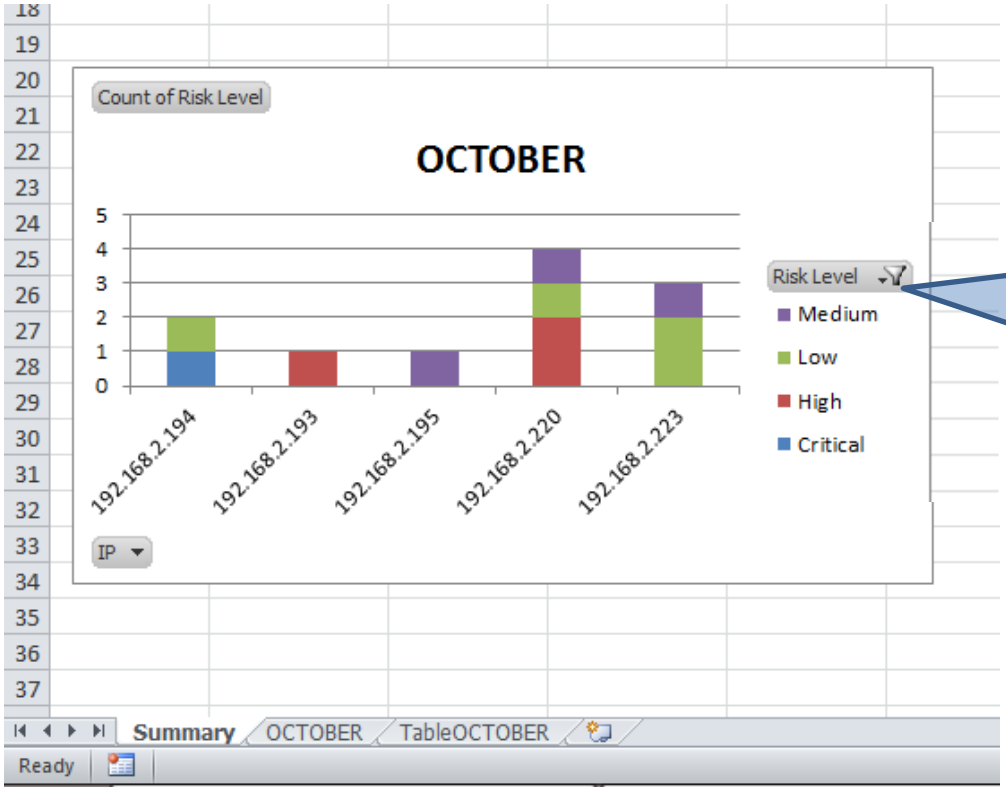


Figure 13 - Top ten infracts on Summary sheet



Filtering controls are available to adjust chart based on data selected

Figure 14 - Chart displaying hosts' vulnerabilities by risk level

When new reports are added to the Summary tab, the new lists of hosts are added to the right of the existing ones. To demonstrate this, a new spreadsheet has been added for the month of November along with the corresponding report (See Figure 15 below).

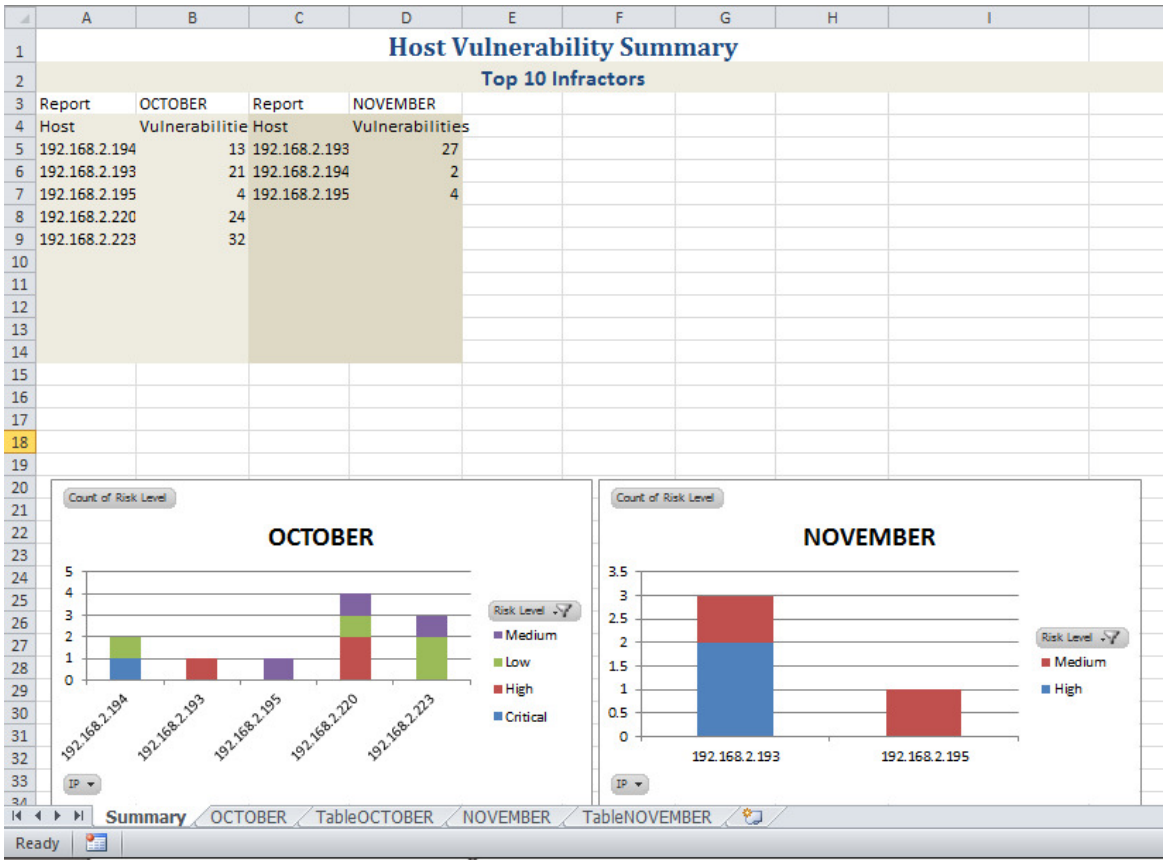


Figure 15 - Summary sheet with November's information

As it can be determined by the graph, only two hosts are present due to the applied filter. However, a complete copulation of hosts can be observed on the top section. In this case, the November report only contained information for three computers.

In the event that additional information is added to a sheet after a report has been created, users can choose to re-generate the report by clicking on the Create Report Button and choosing the spreadsheet whose data has been appended. Users are then prompted to confirm whether they would like to re-generate the report. For demonstration purposes I have appended the Windows Vista scan to November's sheet and re-generated the report (See Figures 16). When the report is re-generated, the new list of top ten hosts and the new chart replace the old ones (See Figure 17 below)

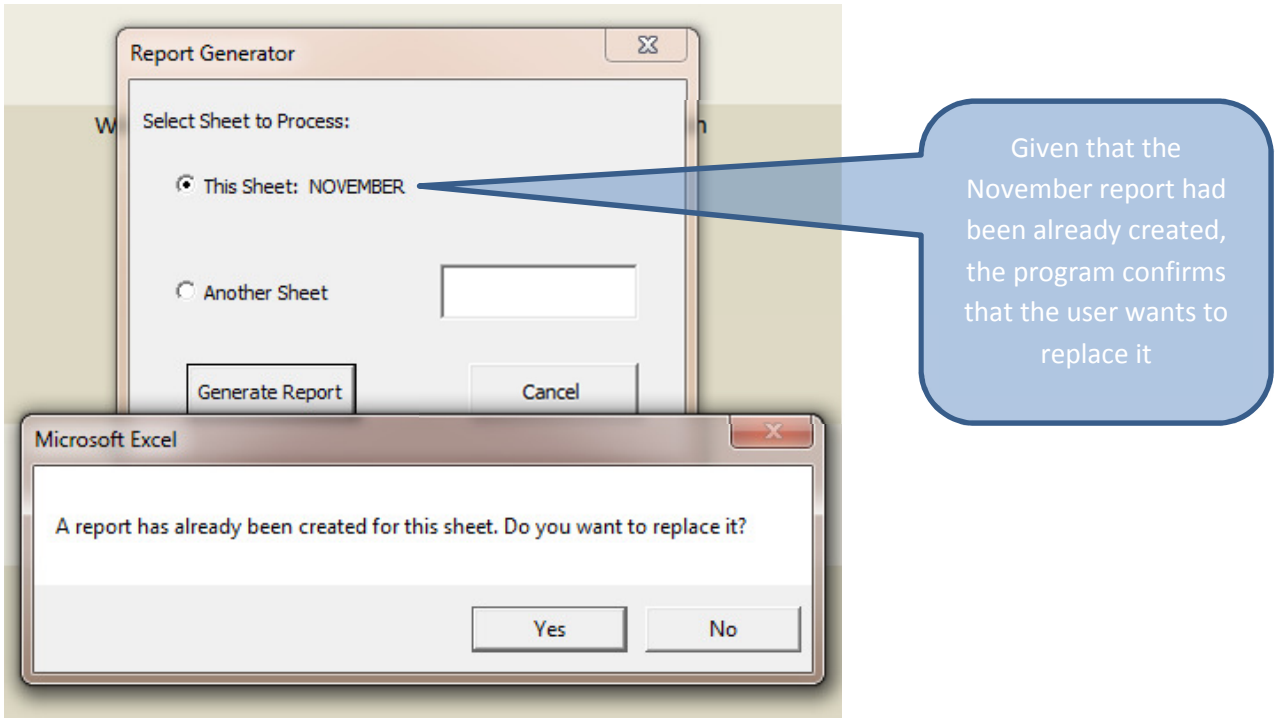


Figure 16 - Re-generation of report

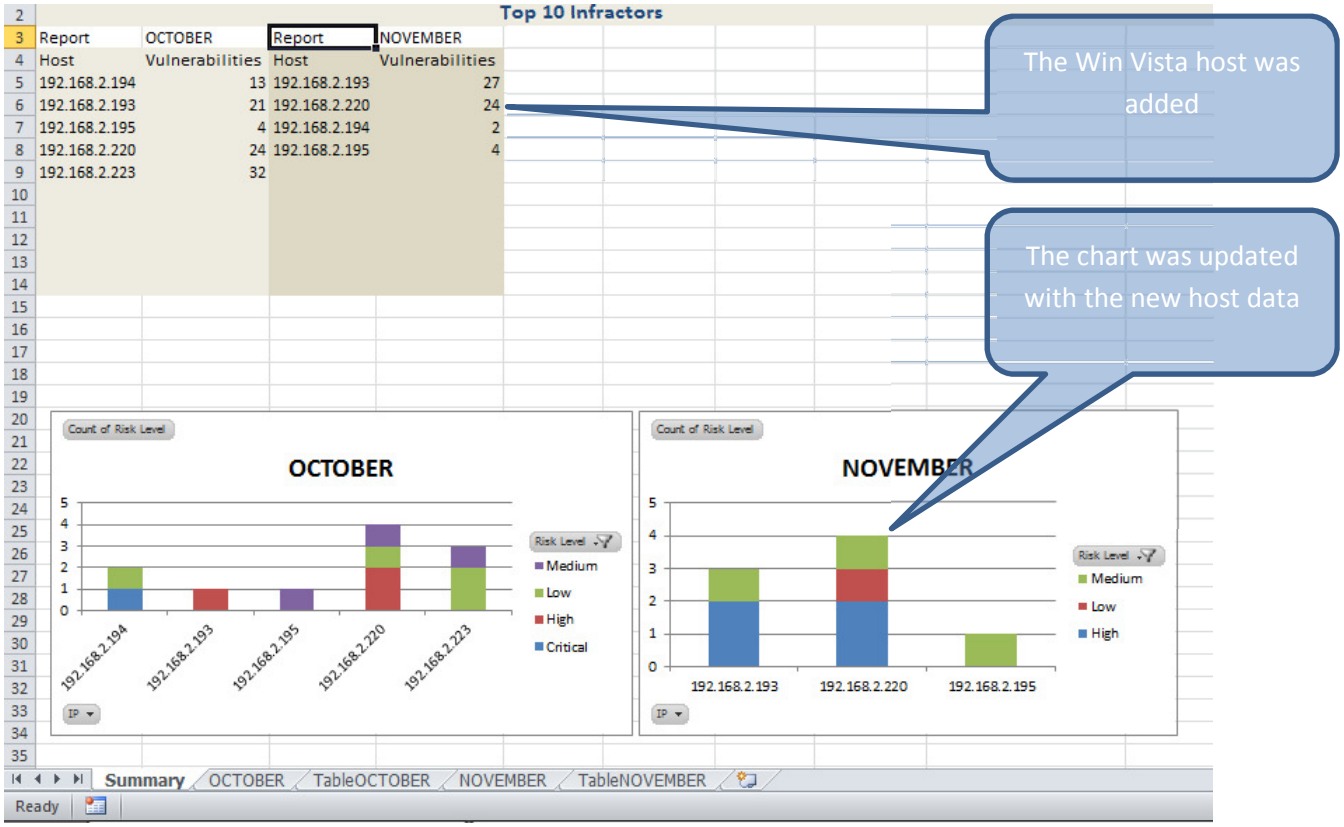


Figure 17 - Updated Summary Sheet

If a third report is generated, the new chart will be created under the two existing charts. For demonstration purposes, a new sheet was created and named Vista to contain the Windows Vista scan results. A new report was generated which resulted in the addition of another top ten list (with only one host as only one machine was scanned) and the corresponding graph (See Figure 18).

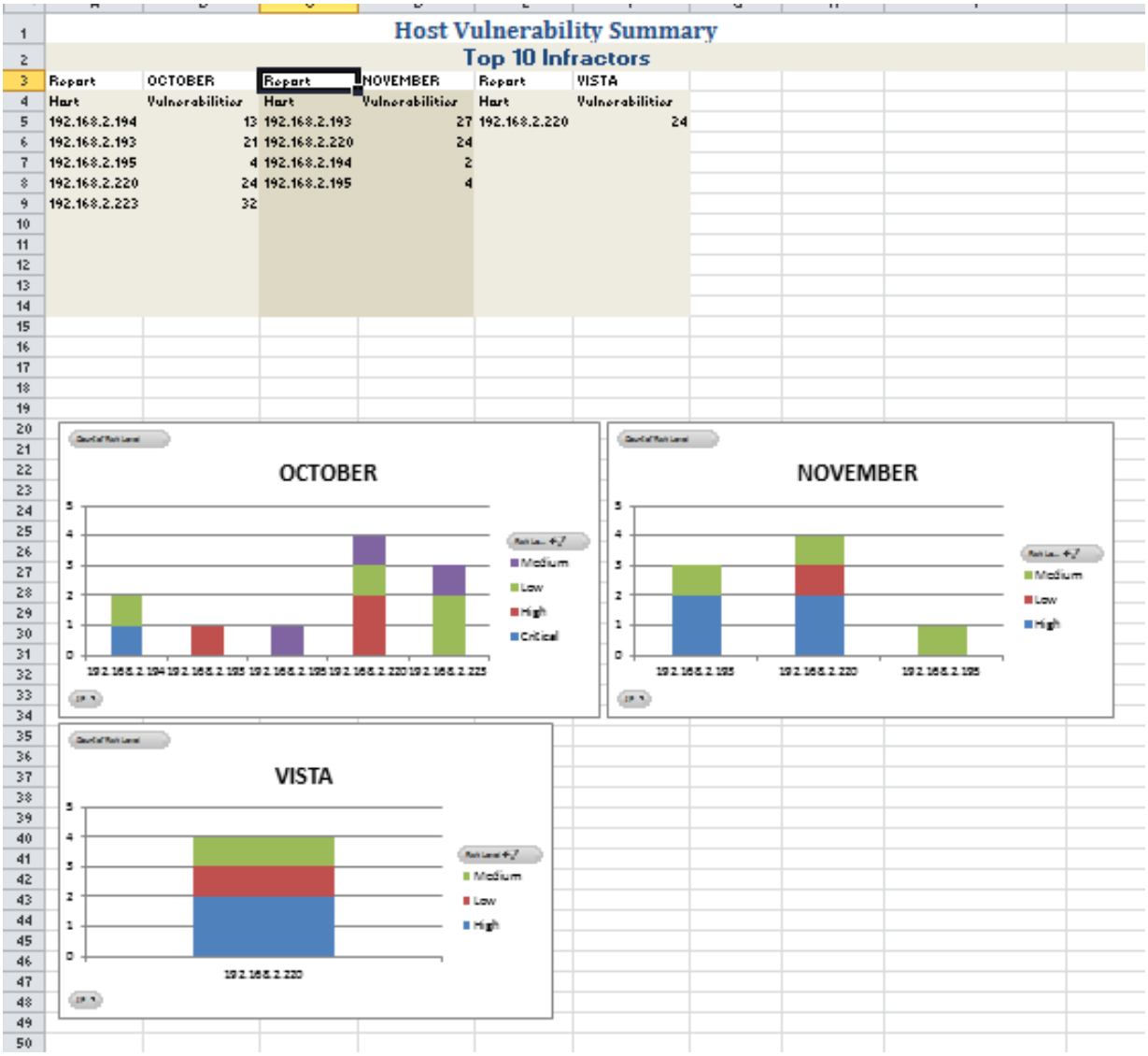


Figure 18 - Summary sheet with report on Vista machine

Content Deletion

If the need arises to delete all charts, or clear the content of the Summary sheet, or delete all the existing tabs to prepare a new sheet for a new set of reports, this can be easily attained by using the buttons in the Vulnerability Reports tab within the Delete Reports group.

The first button, “Delete Charts” only deletes the charts on the Summary sheet. The Top Ten Infracts list is left intact. Alternatively, the “Clear Summary Report” button deletes the entire contents of the Summary sheet with the exception of rows 1 and 2. For demonstration purposes, the “Delete Charts”

button was selected, followed by the “Clear Summary Report”. Upon the selection of each delete button, an excel message box is displayed to confirm the action (See Figure 19). The results can be observed in Figures 20 and 21 respectively.

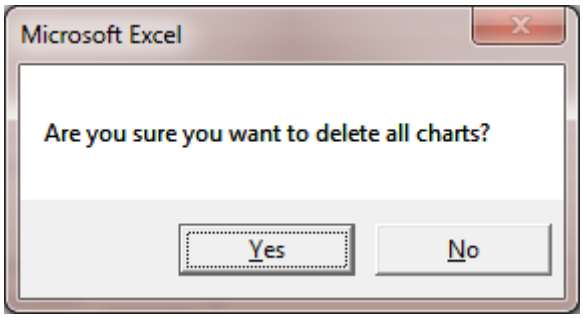


Figure 19 - Confirmation message prior to deletion

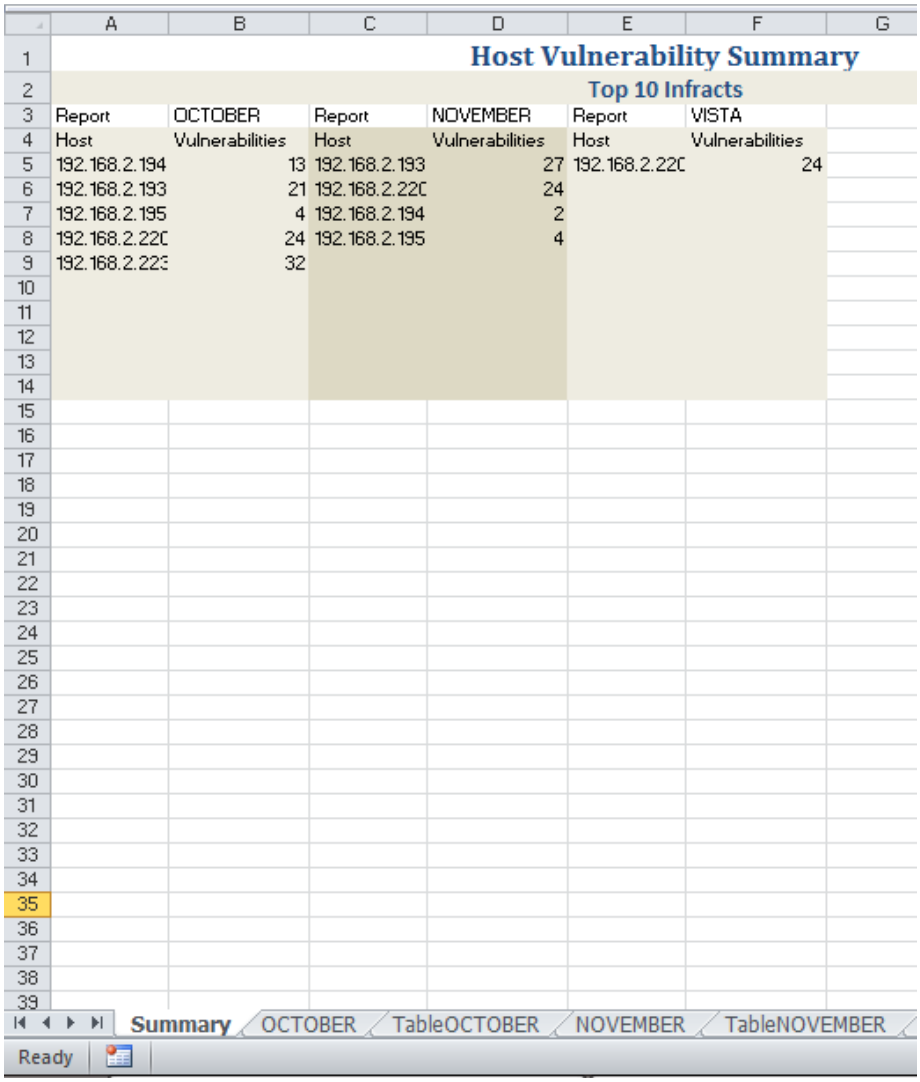


Figure 20 - Summary sheet without charts

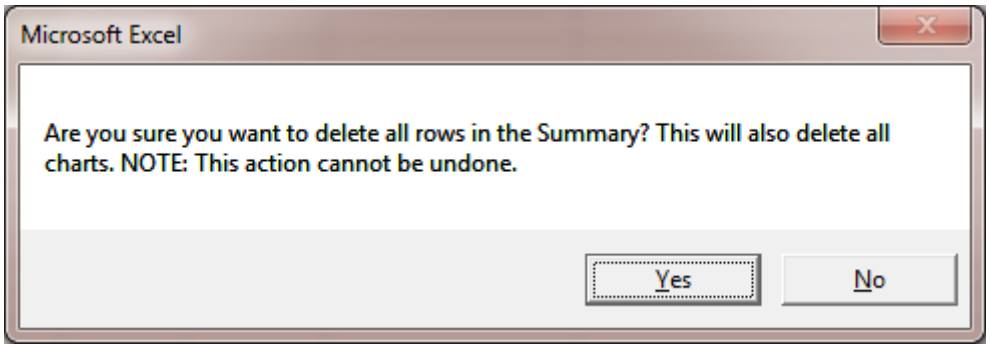


Figure 21 - Confirmation message prior to deleting content of Summary sheet

Finally, the "Delete All Tabs" button automatically deletes all tabs in the workbook except for the Summary sheet. Similarly to the prior two delete buttons, the program issues a confirmation message (See Figure 22).

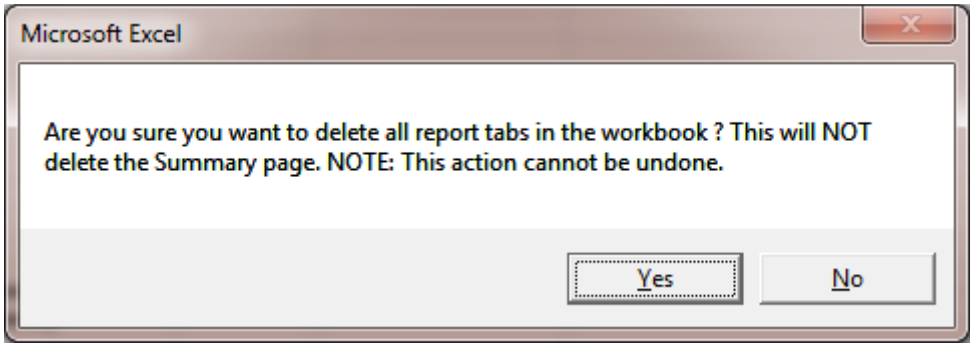


Figure 22 - Confirmation message prior to deleting all sheets in workbook

Discussion

Working on this project helped me internalized several VBA concepts in which I was not very strong, particularly the use of ranges and the way they can be passed from one sub to another. This project was inspired on a spreadsheet I built earlier this year to address the problem of presenting large amounts of data to management. My original spreadsheet did not use VBA but a lot of Excel formulas. When I knew I could do a project of my own I decided to automate the sheet I created.

Some of the initial challenges I faced was that I had to recall all the features of the original sheet by memory. Because, I created the sheet to address a client's problem, I could not keep a personal copy of it. Also, my original sheet received input from several systems. Given that I only have a copy of Nessus, I could only work with one format. However, this proved to be more than enough. The biggest challenge of this project was to Figure out how to successfully import all the data from the HTML. As I mentioned earlier in the document, I decided to use only two scan policies. However, even though the report for both policies was generated the same format (Detailed HTML Report by finding), the HTML

structure was not exactly the same. As a result, I faced the situation of working on obtaining data for one report and get it to work just to find out that my solution would not work for the second one.

I estimate that over 16 hours were spent in the importing of the report to make it work flawlessly. Given that other systems such as McAfee Foundstone do export their reports in a .csv format, I am confident that the import process will go a lot smoother. This is a feature I would really like to add, particularly because it is likely that in my industry I will find the need for a consolidated reporting tool for all the scan results.

The second large challenge I faced was the selection of variable ranges. I had to think very carefully how the different components of the code interact one with another. I learned what works and what does not when it comes to ranges. I realized that ranges have great power but also a learning curve. Once I realize that a range can be declared by using a string first to obtain the address of a group of cells my life was easier. Unfortunately, it was close to 10:00 PM on the due date when I discovered this.

Another area that caused me some trouble was the automation of Pivot Tables. In order to create the top ten hosts with vulnerabilities and their charts I originally decided to copy the needed cells (using ranges) and then make the lists and charts from the selection. However, this proved to be too complex and unnecessary as I could easily do both things directly from the original Pivot Table. It is true that if the table is changed the charts will reflect it, but I think this is more a feature than a disadvantage.

I was particularly happy when I found out by myself how to automate the automatic chart filtering. However, I was only able to filter by existing items. For example, if a data set did not contain any Low risk vulnerabilities, when I tried to filter them it would create an error. I tried to Figure out a way to detect in advance whether a chart contained Low vulnerabilities in order to create a conditional statement that would basically skip the filtering if none was found. I worked on it for a while and looked online for assistance with no avail. Given that I was late on my submission I could not ask Dr. Allen for help.

Overall, this was a strong learning experience. Despite the challenges it presented to me, I was able to get the functionality I needed. In my original proposal I was very ambitious. However, upon talking to Dr. Allen, I was able to scope down some of the original features to a level that was acceptable to him. In the future I would like to continue working on this so I can add those features. One of them is to perform a comparative analysis between the same machines with the same vulnerability through time. I was able to do that for my original spreadsheet and it was one of the most valuable features as management was able to have a clear picture of those machines that were lagging on patches.

I appreciate your time and effort in teaching us such a valuable tool. I feel more empowered as a professional by knowing all the things I can do to automate Excel and add value to my employer and/or my clients. Thank you.