GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, and Incident Handling (Security 504)"
at http://www.giac.org/registration/gcih

Hunting through Log Data with Excel

GIAC GCIH Gold Certification

Author: Greg Lalla, greg.lalla@mail.com
Advisor: Christopher Walker, CISSP, GWEB, GCED, CCISO, GCUX, GCWN, GSEC
Accepted: March 25, 2017

Abstract

Gathering and analyzing data during an incident can be a long and tedious process. The vast amounts of data involved in even a single system intrusion can be overwhelming. Larger and well-funded incident response teams typically have a Security Information and Event Management (SIEM) product at their disposal to help the responder sift through this data to find artifacts relevant to the intrusion. This paper will demonstrate to the reader how to use Microsoft Excel and some of its more advanced features during an intrusion if a SIEM or similar product is not available to the incident responder.

1. Introduction

This document will show you how to use Microsoft Excel to search through dissimilar data to find significant artifacts needed to respond to an intrusion. It may not be ideal to import thousands of log entries into Excel and root through them for a few nuggets that might give you insight into the adversary's exploitation of a network. However, if a SIEM product is not running in the enterprise, with a little knowledge and a small amount of coding, you can use Excel as a suitable substitute. There will still be a manual effort involved as there is no magic button that will produce all the artifacts that you wish to find, but the process described here will make it easier and more intuitive to filter out all the unwanted data.

When examining logs, you will look for indicators of compromise (IOCs) that can point you in the direction of other compromised systems. The examination of logs is not a deep dive forensic type of analysis. During an incident, there is not enough time to look at every log entry from every system. The method shown below will help find the obvious artifacts and identify the next system to examine.

This guide will contain up to three methods for each example presented. First, the paper will show some of the things you can do with Excel by just using the toolbar commands. Second, if available, an Excel Function will be created to show how it can be slightly automated. Third, to enhance the Excel Function process even further, Visual Basic for Applications (VBA) code will be provided. Knowing alternate ways of manipulating different types of data will allow you to incorporate the results into the standard output described below.

To prevent this paper from being overly long and difficult to search through, the VBA code will be made available on the GitHub website to make it easier to replicate. The GitHub URL is https://github.com/gregory-lalla/GCIH_Gold/.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

2. Requirements

2.1.   Excel Version

The Excel Functions and VBA code in this paper were written and executed using Microsoft Excel 2010. Other versions of Microsoft Excel based on the Office Open XML (OOXML) specification (Excel 2007 and later) should have most of the same functionality. There are exceptions, such as "Making a Macro that changes the cell colors and making changes to other aspects of cells may not be backward compatible" ("Microsoft Excel", 2016). Also, the location of some options may be in different menus or locations within a menu.

2.2.   Developer Toolbar

To use the techniques described in this document, you will need to have the Developer Toolbar added to the Ribbon. For instructions on how to enable the Developer Toolbar, visit https://msdn.microsoft.com/en-us/library/bb608625.aspx/.

3. Organizational Concepts

Formatting, filtering, and organization are the core techniques in this paper for finding relevant information needed to respond to an incident. The following are suggestions on how to get Excel to display the data in a way that is easy to analyze. We will use these techniques when looking at each of the different types of logs discussed later in this paper.

3.1.   System Time

First, depending on the geographic location of your systems, the time zone settings may need to be adjusted. If the location of all the systems is in the same time zone, then you may want to perform all your data correlation in the local time zone. If your systems span time zones, it is best to do all the analysis using the Coordinated Universal Time (UTC) time zone. Using UTC, all the data will line up chronologically. To make the process easier, the system time should be changed to UTC so that applications that use the computer's time to display the timestamp will automatically produce the correct time format.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Another setting to change is the Date and Time format of your analysis computer. Excel uses the Date/Time format of the system when it displays the information in the formula bar. To keep things consistent throughout the investigation and within reporting, have the Date/Time in the 'mm/dd/yyyy hh:mm:ss' format (without the quotes). This setting makes all the date and time values 19 characters long. To make this change, follow these instructions: https://support.office.com/en-us/article/Change-the-Windows-regional-settings-to-modify-the-appearance-of-some-data-types-edf41006-f6e2-4360-bc1b-30e9e8a54989/.

Finally, the dates and times in the Excel spreadsheets need to be formatted to display the same 'mm/dd/yyyy hh:mm:ss' format. In the below example, you change the Date/Time format by selecting Column A which contains the date and time values, then right clicking the column and selecting 'Format Cells…'. This process should bring up a new window titled 'Format Cells' with the 'Number' tab already selected. Under 'Category,' click 'Custom.' In dialogue box labeled 'Type:' enter mm/dd/yyyy hh:mm:ss (Figure 1).
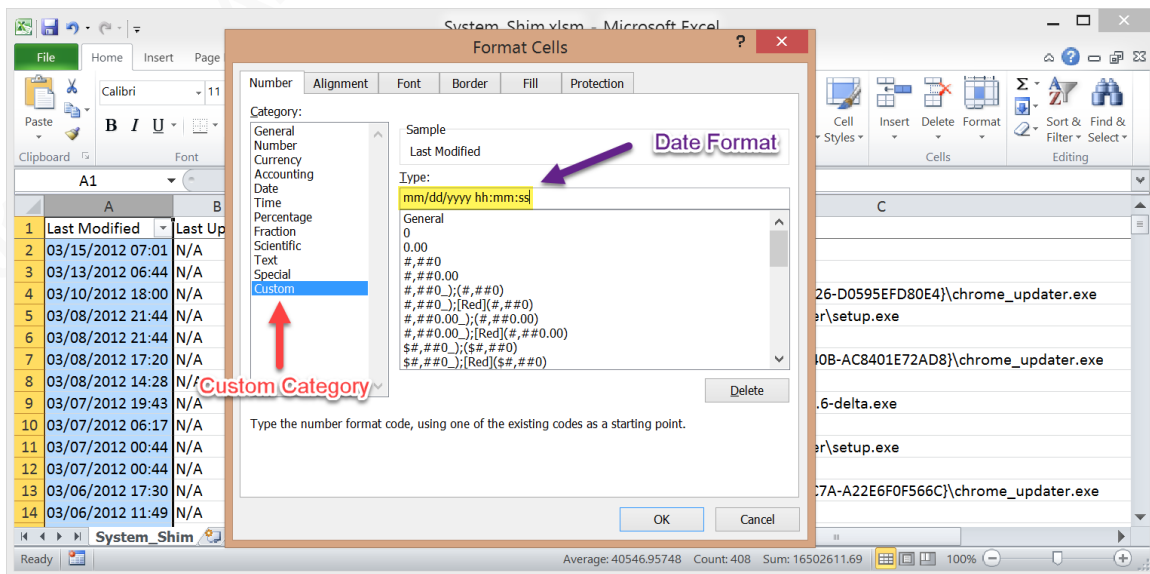
Figure 1. Date Format (Lee, 2014, digital case files)

3.2.   Consistent Results

Other columns displaying data should also have a uniform appearance, which makes it easier to spot trends, inconsistencies, patterns, etc. In this paper, the following column headers will be used across all spreadsheets to achieve that consistent look:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

- Column A Header Name = Date/Time
- Column B Header Name = Account
- Column C Header Name = Computer
- Column D Header Name = Description
- Column E Header Name = Details
- Column F Header Name = Properties
- Column G Header Name = Miscellaneous
- Column H Header Name = Artifact

Note that it is acceptable to have empty columns to the spreadsheet if there is not enough data to fill all the suggested columns.

3.3.   Formatting

To easily read the data, you will freeze the top row and apply a bold font to it; enable filters on the columns; set the column widths to 'Autofit'; and 'left justify' the entire spreadsheet. Reduce the data where possible and sort it by the Date/Time column from oldest to newest. Please see the GitHub page which has an Excel template of the standard format named 'Standard_Format.xltm' (https://github.com/gregory-lalla/GCIH_Gold/blob/master/Docs/Template/Standard_Format.xltm).

3.4.   Keywords, Named Cells, and Filters

One of the common themes repeated in this paper is data reduction. Excel can handle a lot of data, but there is a cost in the time it takes to manipulate and analyze that data. Each log file examined will have unique entries which the responder must understand to reduce the data without losing critical items relevant to the incident.

One method of data reduction is the use of keywords. A list of keywords will be used to help pinpoint known or suspicious activity related to the intrusion. There should be two types of keyword lists maintained. One master list of all the terms discovered during the throughout the investigation and an event list for each type of log analyzed. Tailor the event keyword lists to the log files you are inspecting to minimize the output of the filtered data.

When reducing data, one issue faced is the location of the data you want to keep changes as rows and columns get adjusted. This shifting of data needs to be kept in mind when working with 'Named Cells' (Blue arrow in Figure 2) as the properties assigned to

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

the cell stay with that cell, even if the data in the cell moves to a new location. Therefore, reduce and adjust all the data before using 'Named Cells.' If the data in a 'Named Cell' will change, a good way to keep track of that data is to use 'Fill Colors.' When a keyword search gets a match, highlight the cell or entire row with a particular color. Using filters, you can locate the 'colored' data with a few clicks of the mouse.

To manually work with keywords, the basic search feature can be used to find each instance of the keyword. Enter a unique identifier in the cell 'Name Box' (Blue arrow in Figure 2) when a keyword is found in a cell. The unique identifier name has the following rules: "The first character of a name must be a letter, an underscore character (_), or a backslash (\). Remaining characters in the name can be letters, numbers, periods, and underscore characters" ("Define and use names in formulas – Excel", 2017).
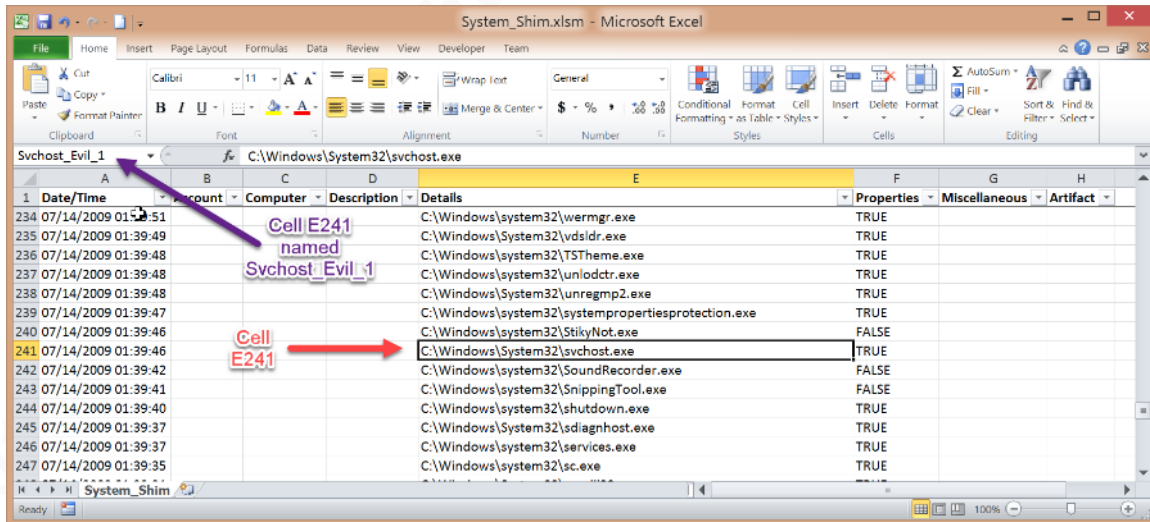
Figure 2. Named Cell. (Lee, 2014, digital case files)

After naming the cell, fill the entire row that contains the keyword with a color (Figure 3). For each unique keyword found you can continue to use the same color or change them to different colors distinctively associated with each keyword. Using the fill colors is helpful when using the filter tool, which will be discussed and demonstrated further into the paper.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
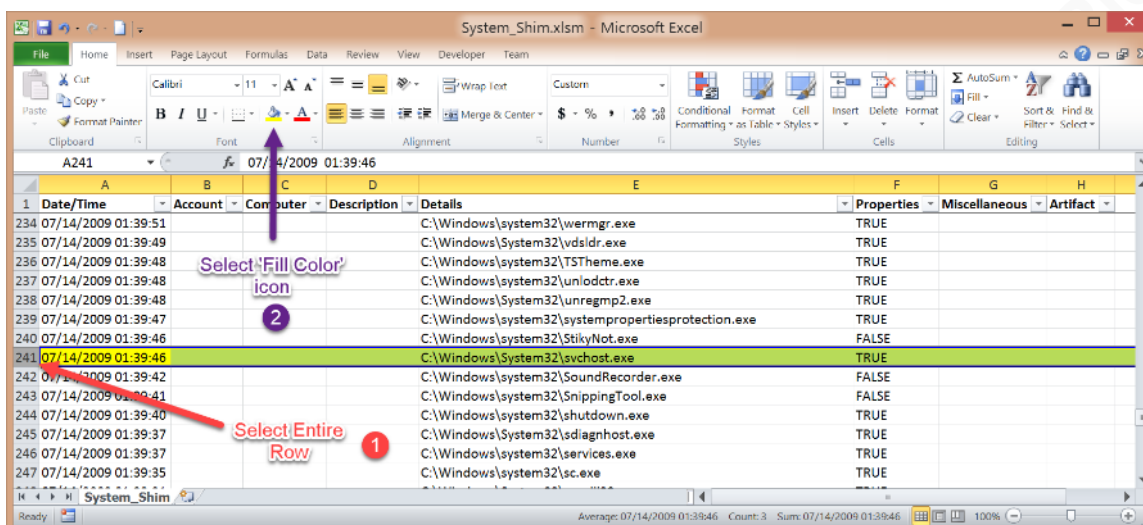
Figure 3. Fill Color. (Lee, 2014, digital case files)

Click 'Find Next' in the 'Find and Replace' window to find the next instance of the keyword. Once found, give that cell a similar, but unique name to distinguish it from the first keyword found and give it a 'Fill Color.' As an example, the first unique cell Name would be 'Svchost_Evil_1' and the second would be 'Svchost_Evil_2' for hits found on keyword 'svchost.exe.' After all the keywords are searched and found in this manner, you can navigate to those cells by clicking the dropdown arrow in the cell 'Name Box' and selecting one of the entries (Figure 4). You can also manage the 'Named Cells' and see their location by going to the 'Formulas' ribbon tab and selecting the 'Name Manager' icon (Figure 5).
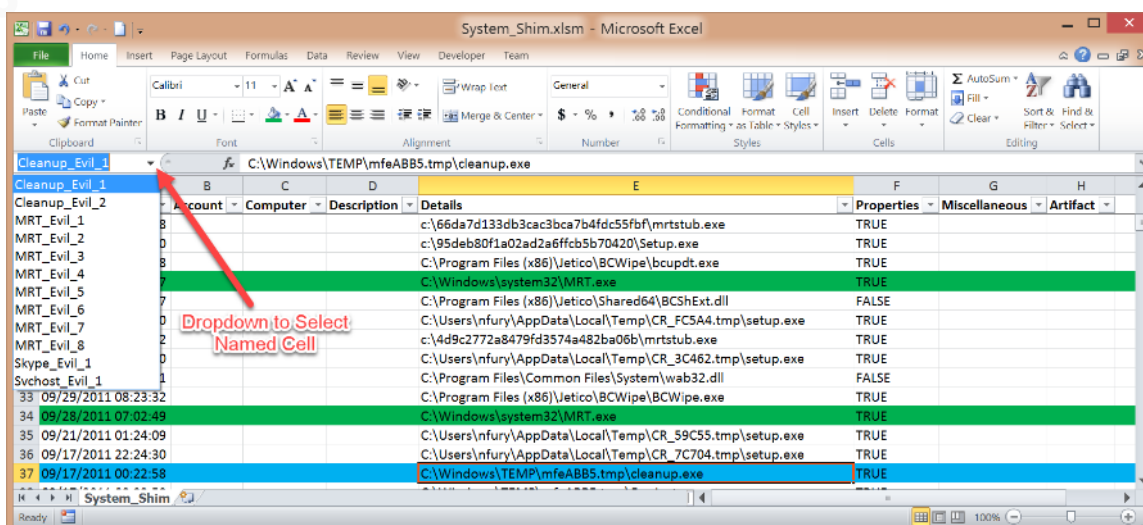
Figure 4. Selecting Named Cells. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
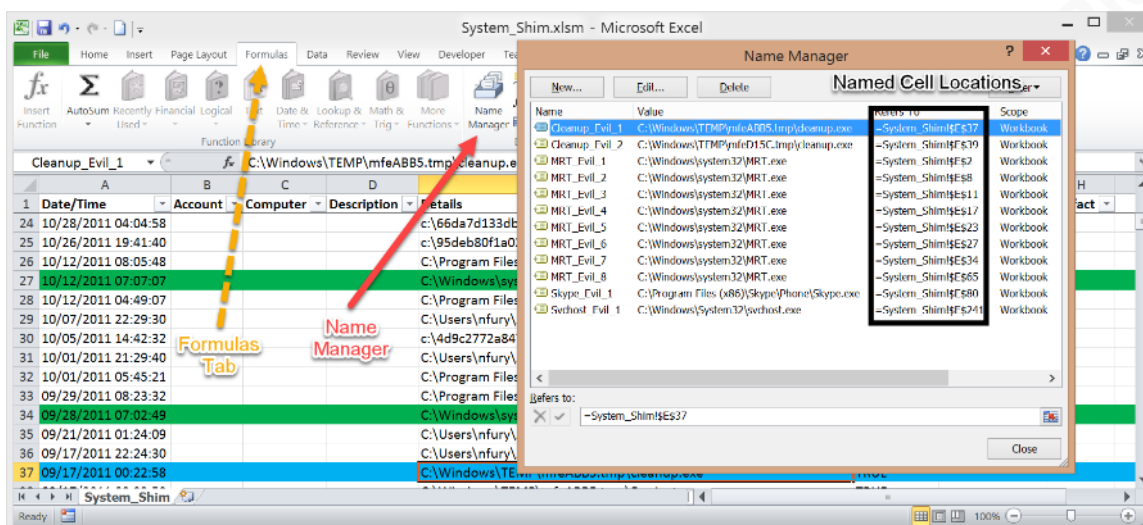
Figure 5. Name Manager. (Lee, 2014, digital case files)

Using 'Named Cells' is a quick way to locate known artifacts. It is also an excellent way to find more keywords by examining the data around those cells.

Filters are another technique to find data and visually reduced the data to only show those cells that the analyst wishes to view. These are particularly helpful when you know you'll have many hits on your keywords and would like to see them without the clutter of all the other rows not associated with those keywords. There are two ways to use the filters. There is a basic filter where you can have up to two items filtered per column using AND OR operators. To filter on more than two items, there are Advanced Filters which can be used to search for data with more complex options.

We enabled basic filters already in an earlier example. To access them, on the column we are looking to search, click on the dropdown arrow in the header cell. Select the 'Text Filters' option and then choose one of the filter selections (See Figure 6).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
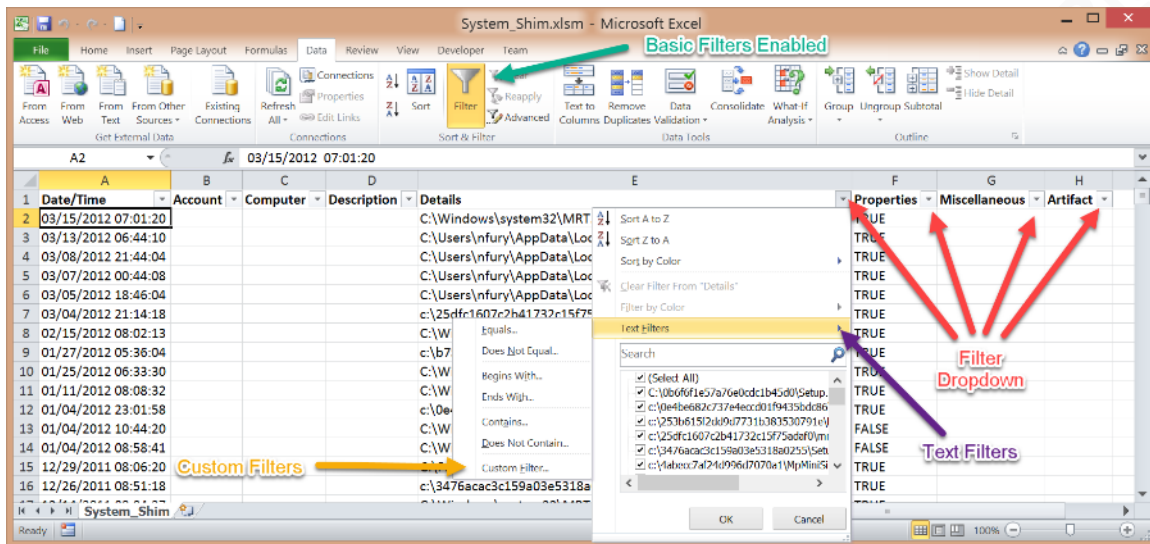
Figure 6. Filter Dropdown Menu. (Lee, 2014, digital case files)

When the filter window comes up, enter the word(s) you want to locate (or exclude) in the dialogue box. The filter will display only the rows that have (or don't have) the words in the column.

You can also search on the 'Fill Colors' used during the keyword search to filter only on those colors you wish to see (Figure 7).
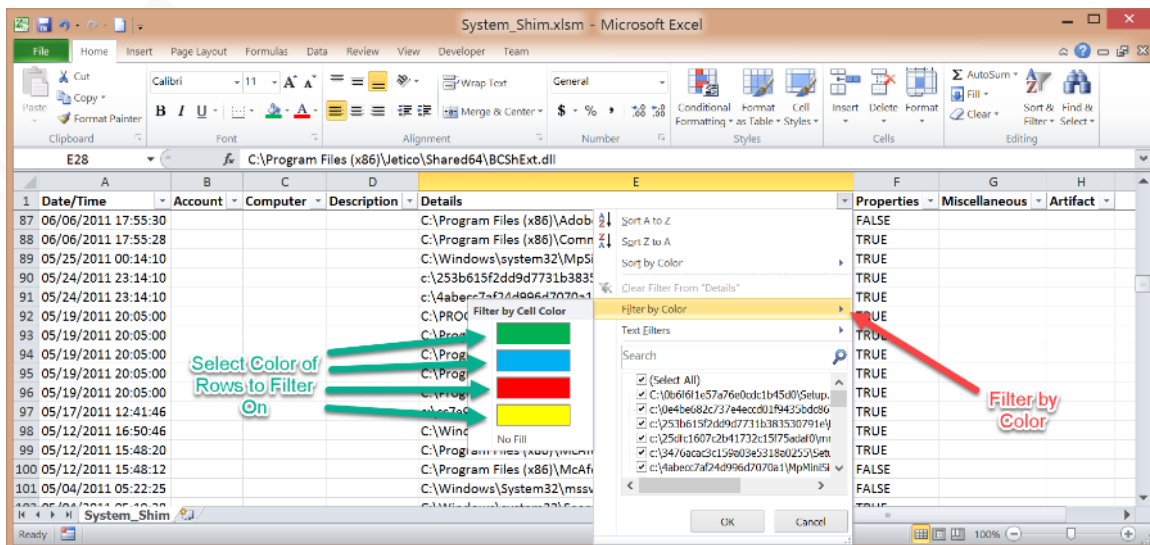
Figure 7. Search on Fill Color. (Lee, 2014, digital case files)

The advanced filters allow you to search for more than two keywords in a column. You enter the keywords into unused cells in the spreadsheet. The first cell is the column header name of the column to search. The cells underneath are the search terms.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Words listed vertically use the OR operator and those listed horizontally use the AND operator. Figure 8 shows an example of an advanced filter, where you are searching the 'Details' column for five keywords. Surrounding the keyword with asterisks finds cells that CONTAIN the word. For details on filters, see https://support.office.com/en-us/article/Filter-by-using-advanced-criteria-4c9222fe-8529-4cd7-a898-3f16abdff32b/.
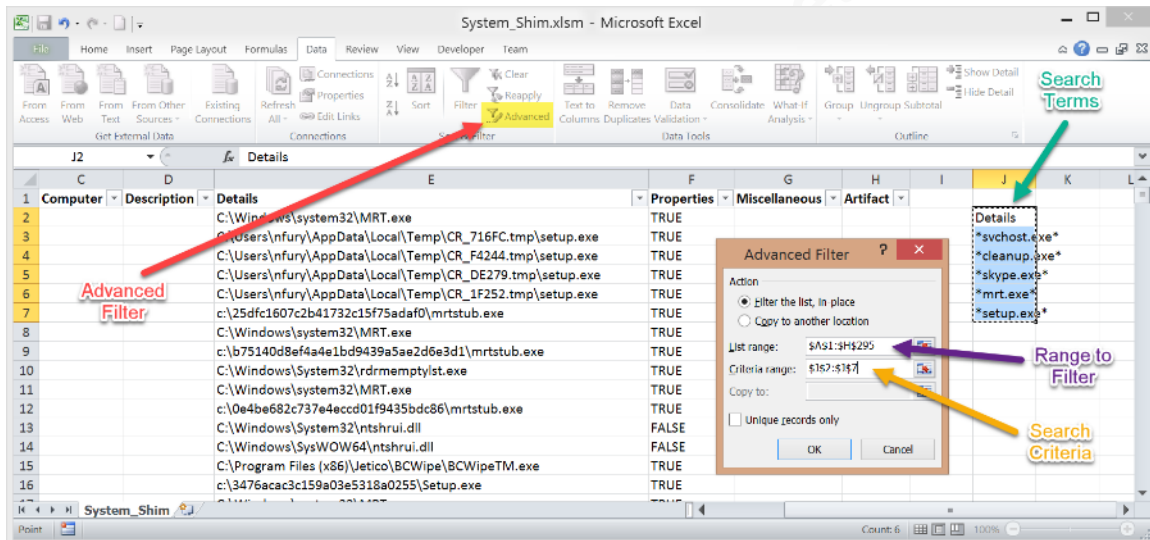
Figure 8. Advanced Filter. (Lee, 2014, digital case files)

Note the timeframe of the filtered keyword data. When looking at all the log files with tens of thousands of lines of data, this range will be important when deciding what to export for analysis (if given the option). The dropdown menu on the Date/Time column is used to filter by timeframe (Figure 9). Selecting the 'Between…' filter will allow you to capture all the data between two dates narrowing down what data you need to examine.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
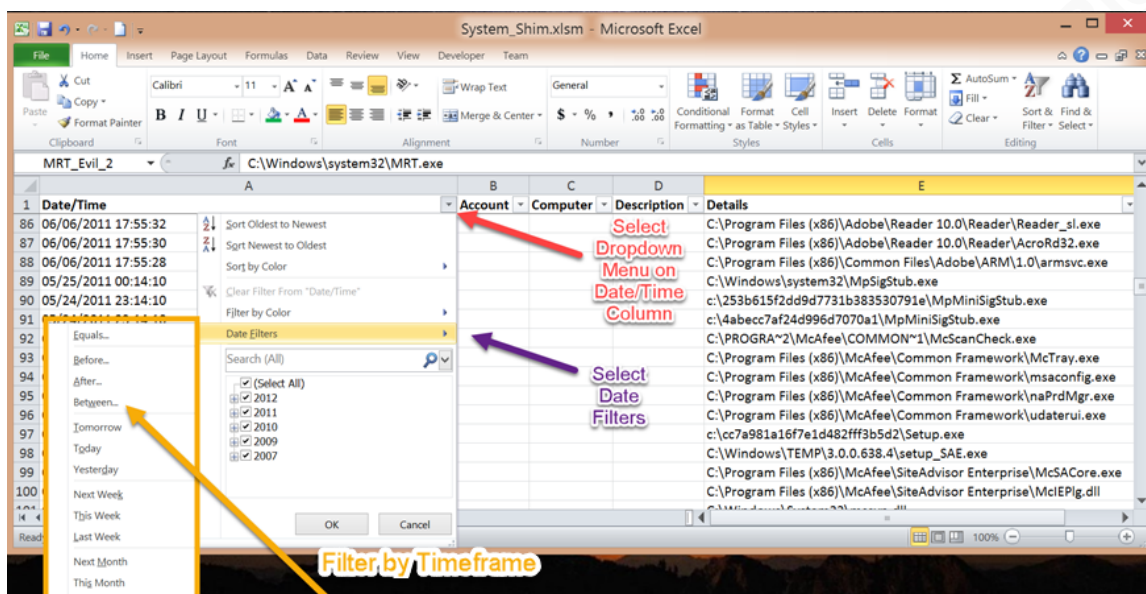
Figure 9. Search by Date/Time. (Lee, 2014, digital case files)

One thing to be aware of when filtering by timeframe is that artifacts may have timestamps that occur outside of the incident timeline. These timestamps can happen when the artifact is showing the file compile time instead of its execution time. Another example would be time-stomping, which is an anti-forensics technique to change the MFT timestamps of a file.

Any new entries found that are related to the intrusion should be 'Named' and 'Filled.' You should then copy off the highlighted rows onto a separate workbook which will contain all the rows of interest from all the logs collected.

3.5.   Macros

Many of the tasks described above are tedious and repetitive. In such circumstances, it is easier to automate the tasks using VBA macros. You'll be provided with several VBA macros so you do not need to perform the task manually. To use macros, open the exported log file in Excel and then save it as an Excel macro-enabled workbook file with an XLSM file extension. Open the newly created .XLSM file to import the macro into the spreadsheet. Once the spreadsheet is open, you need to launch the VBA Editor window by hitting the Alt-F11 keys. With the VBA window open, select the 'File' dropdown menu and click on 'Import File.' Browse to the .BAS file with the code you are looking to run and select 'Open.' Close the VBA window and head back to

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

the spreadsheet. Under the 'Developer' tab, select the 'Macros' button, highlight the newly imported macro in the popup window and select 'Run' (See Figure 10).
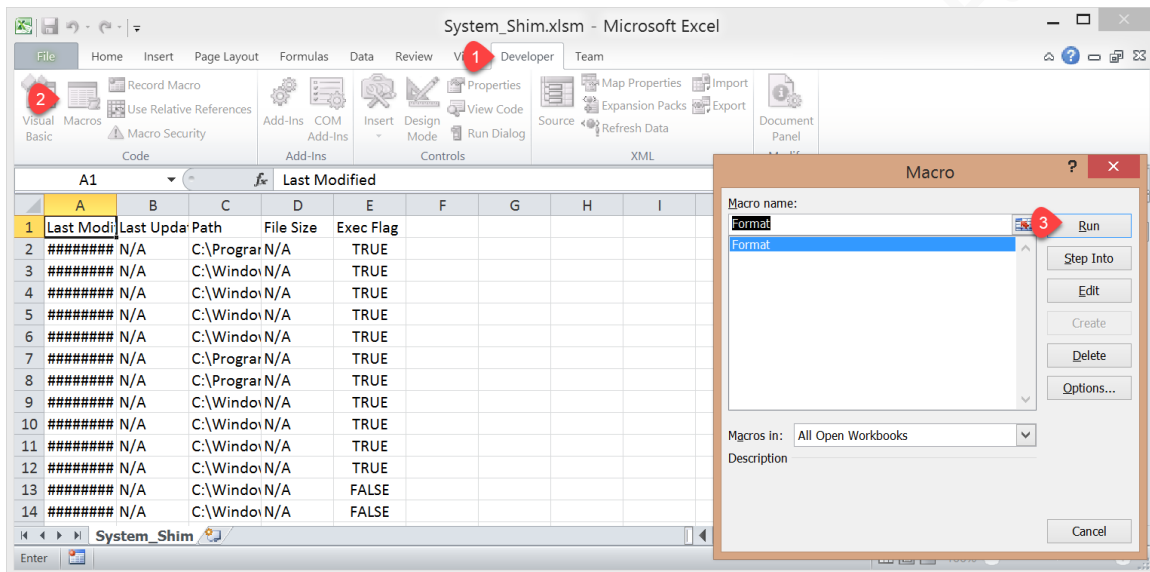
Figure 10. Run Macro. (Lee, 2014, digital case files)

There are several macros in the GitHub repository that can automate the processes described above. See Appendix C for a list of macros available in the repository and a description of what each one accomplishes.

3.6.   Pivot Tables

Pivot Tables are an excellent way to sort data in a way that visually allows you to pick out artifacts of interest. "A pivot table allows you to create an interactive view of your dataset. With a pivot table report, you can quickly and easily categorize your data into meaningful information, and perform a wide variety of calculations in a fraction of the time it takes by hand" (Jelen, 2006, p. 9). This categorization of data is particularly true when working with Windows Event Viewer logs. Exporting the data in XML table format provides headers that can be used to categorize the data in ways that will allow you to see trends and commonalities. The GitHub site has a document named "Pivot_Table_Example.docx" that shows examples of how you can work with pivot tables to analyze data (https://github.com/gregory-lalla/GCIH_Gold/blob/master/Docs/Supplement/ Pivot_Table_Example.docx).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

4. Gathering the Data

Having explained the essentials of organizing logs with Excel, real understanding comes from seeing the techniques demonstrated on specific types of log files. This section will discuss examples of different log files that an incident responder may have to examine while investigating a compromised network. Then in the following section, a case study will be analyzed employing the techniques described in this paper. Appendix D provides a detailed explanation on how the Windows log files in the case study were formatted. A supplement document on the GitHub site named 'Additional_Log_Formatting_Instructions.docx' (https://github.com/gregory-lalla/GCIH_Gold/blob/master/Docs/Supplement/Additional_Log_Formatting_Instructions.docx) provides a detailed explanation of the remaining log files mentioned in this section.

4.1.   Windows Logs

4.1.1. Event Viewer Logs

One of the primary sources of information/data gathered from a Windows operating system will come from Event Viewer logs, especially if recommended auditing settings are configured appropriately (see https://technet.microsoft.com/en-us/library/ee513968%28WS.10%29.aspx for recommendations). Unfortunately, a "Microsoft Windows event log is a binary file that consists of special records – Windows events" ("Windows event log essentials", 2017) and parsing the data is not as simple as manipulating the file in a text editor. Also, when exporting the data from the native Windows Event Viewer tool, depending on the format chosen, different data is returned. Last, if you've ever tried to filter or search using the native Windows Event Viewer tool, you know that it has many limitations and is extremely slow.

Because Event Viewer logs can contain hundreds of thousands of entries, it is essential to reduce the data to a manageable level. Since the Event Viewer GUI is extremely slow, the native Windows command line utility named WEVTUTIL.EXE should be used to export the data. This tool comes with its challenges such as producing XML files that Excel cannot open. When running the command, there are several options

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

to massage the data to a format that Excel accepts. To get the correct results, you may need to mix and match the switches until you get a compatible output.

4.1.2. ShimCache Entries

The ShimCache or Application Compatibility Cache entries show executables that were likely run on the system. "The Windows Shimcache was created by Microsoft beginning in Windows XP to track compatibility issues with executed programs…It is important to understand there may be entries in the Shimcache that were not actually executed" (Parisi, 2015). "Microsoft designed the Shimcache in Windows Vista, 7, Server 2008 and Server 2012 to incorporate a 'Process Execution Flag' category for each entry" (Parisi, 2015). For further details on the Shimcache and the differences between XP/2003 and Vista+, see https://www.fireeye.com/blog/threat-research/2015/06/caching_out_the_val.html

4.1.3. Shellbags

Shellbags reflect the locations that the user has traversed using Windows Explorer. "Shellbags are found in the Windows Registry and store user preferences for folder display in Windows Explorer, such as the size of the window or how items were listed. For a folder to exist in the shellbags, it must have been opened in Windows Explorer at least one time by the user" (Cowen, 2013, ch. 13). The user's usrclass.dat (Vista+) and NTUSER.DAT registry files contain the Shellbag artifacts.

4.1.4. AutoRun Entries

Autorun entries refer to "software that runs automatically without being intentionally started by the user. These include drivers and services that start when the computer boots; application, utilities and shell extensions that start when a user logs on; and browser extensions that load when Internet Explorer is started" (Russinovich, 2011, ch. 5). To parse autorun data, run the autorunsc.exe command against the registry files of a system. You can run the tool on a live system or offline by mounting an image of the hard drive and pointing the program to the newly mapped location.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

4.1.5. Web Browser Logs

The client browser may provide clues on how the adversary initially got onto a system, either through the user browsing the internet or by clicking on a link in an email or document. That information may be quite valuable in targeted attacks against the employees of a company. This section will examine Internet Explorer (IE) history/cache logs in index.dat files (IE 9 and below). Newer versions of IE browsers store their data in an Extensible Storage Engine (ESE) database. There are still other browsers that use SQLite databases. However, the data represented should still be similar once exported out of the database and into a plaintext format.

4.1.6. MFT Entries

"The Master File Table (MFT) is the heart of NTFS because it contains the information about all files and directories. Every file and directory has at least one entry in the table, and the entries by themselves are very simple" (Carrier, 2005, p. 274).

4.1.7. Prefetch Entries

"Application prefetch is intended to enable a better user experience within Windows systems by monitoring an application as it's launched, and then 'prefetching' the necessary code to a single location so that the next time the application is launched, it launches faster. This way, the system doesn't have to seek across the file system for DLLs and other data that it needs to start the application – it knows where to find it" (Carvey, 2014, p. 98).

4.2.   Other Log File Types

4.2.1. Linux System Logs

The formatting of logs produced by many applications and services in Linux make importing the data into Excel challenging. The manual massaging of the data is not difficult but must be done in several stages using a variety of tools. Many of the tools originated on the UNIX operating system, but the ones used in this document have all been ported over to Windows. These tools are from the GNU utilities run under the Open Source tools package CYGWIN.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Linux produces several plain text system logs that may have value to the incident response analyst and which all have the same log format. In the supplement, we will look specifically at the Syslog, Auth.log and Cron.log files, but we can use the same techniques against the Daemon.log, Boot.log, Mail.log and other system log files. See http://www.thegeekstuff.com/2011/08/linux-var-log-files/ for a listing of system logs that may appear on a Linux host.

The syslog daemon handles messages from the entire system to include many of the system logs mentioned above. Depending on the configuration of the logging in the syslog.conf configuration file, the bulk of the message usually are sent to the 'syslog' (often named 'messages') log file. The Auth.log file contains user authentication information, and the Cron logs record the activity of the cron jobs (scheduled tasks) run on the system.

4.2.2. Apache Access and Error Logs

Another log you may find on a Linux server is the Apache www-access.log file which records connections made from a client to the web services of the system. Web servers are often exploited and could provide the adversary with an initial stepping stone into a network. What gets logged can vary based on the configuration of the web server.

The Apache Error logs contain web server error and resource alerts. The formatting of this log is similarly to the Linux system logs discussed previously. The log should have the dates and delimiters corrected, so each field is in its proper column.

4.2.3. IIS Web and FTP Logs

The native format of the IIS Web log allows for easy importing of data into Excel. However, the scattering of headers throughout a log file presents the only real issue. By filtering on the Date/Time column for entries that are not in the Date/Time format, the extra headings can be found and removed.

4.2.4. IPTables Firewall Logs

Scrutinizing network traffic when combined with other types of artifacts may also be beneficial to your investigation by identifying communications associated with the event and adding those IP addresses to your keywords for further examination.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

IPTables and most other firewalls will produce logs that can be exported and examined. IPTables is run on Unix/Linux systems and has logs similar in format to the Linux system logs discussed earlier in this paper. Therefore, the same steps are used to fix the dates and set the delineation of the fields.

4.2.5. Packet Captures

Packet Captures can be included as well into the analysis of the incident. For the packets to get formatted correctly when exported, the columns in Wireshark need to be changed to produce the results desired.

4.2.6. Snort and Bro IDS Logs

Host-based and network-based IDS logs are critical to incorporate into the analysis as it may be the primary reason you are aware of the intrusion in the first place. These logs usually contain data from the best vantage point, either from the network or host perspective.

Snort is an open source product that "supports sending real-time alerts when an intrusion event is detected and can even be used as an inline 'intrusion prevention system' that enables you to receive alerts in real time and in several different medium, rather than having to continuously sit at a desk monitoring your Snort system 24 hours a day" (Caswell, 2007, ch. 2).

The supplement will look at the logs produced by running snort in Fast alert mode which "writes the alert in a simple format with a timestamp, alert message, source and destination IPs/ports" (Roesch, 2003).

Bro is another open source intrusion detection system. "Bro inspects all traffic flowing into and out of a network. It can operate in passive mode, in which it generates alerts for suspicious activity, or in active mode, in which it injects traffic to disrupt malicious activity … Unlike other NIDSs, Bro monitors traffic flows rather than just matching patterns inside individual packets. This method of operation means that Bro can detect suspicious activity based on who talks to whom, even without matching any particular string or pattern" (Nemeth, 2010, ch. 22). Bro produces several logs, each of which can use the technique described in the supplement to achieve our standard layout.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Here are some of the logs Bro generates: Conn, DHCP, DNS, Files, HTTP, Weird, etc. The supplement will examine the Bro Connect log (conn.log) which records TCP, UDP and ICMP connections.

5. Case Study

5.1. SANS Stark Research Labs

The techniques for extracting and importing the relevant log files in this Case Study are shown in Appendix D. Macros to automate the manipulation of the data are listed in Appendix C. The application of these tools and techniques will be demonstrated by examining an incident presented in the SANS Forensic 508 Advanced Computer Forensic Analysis and Incident Response exercise workbook titled 'Stark Research Labs Intrusion.' The scenario describes how a company received a phone call on April 06, 2012 @ 5 PM EDT from a 3-4 letter government agency stating "We have seen a few hundred megabytes of sensitive data leave your network bound for a foreign country. Don't ask how we know, but you might want to check 10.3.58.7 on your network" (Lee, 2014, ex. 0 p. 2).

Given other information about the company and its assets, a preliminary keyword list is compiled by the incident response team to include the following terms: hydra, star fury, agents, secret and formula. When discussing log files and artifacts below, you can assume that the files have already been imported into Excel with the standard format.

5.1.1. WinXP-TDugan (WKS-WINXP32BIT)

The host reported as leaking data (IP address of 10.3.58.7) is a machine running Windows XP. Since data appears to be actively leaving the host, the first step would be to see what applications have been running on the system. There are two artifacts made available to us that can show what was running on the workstation. The first is the Prefetch files. Applying our initial keyword list against the output gives us one hit on HYDRAKATZ.EXE. The name of the file is similar to Mimikatz, which is a post exploitation tool to capture user credentials. We will flag this suspicious executable by giving the hit a 'Named Cell' of 'HYDRAKATZ _1' and highlighting the row in yellow (Figure 11).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 11. Named Cell and Highlight keyword hits. (Lee, 2014, digital case files)

This keyword hit also gives us an initial timeframe of the incident. The file Hydrakatz ran on 04/03/2012, and a notification was sent to the company regarding a data leak on 04/06/2012. Looking at the Prefetch entries around that timeframe gives us addition clues. Right around hydrakatz.exe, there is a file that appears to be a randomly generated name, PKXEZY1TJI98.EXE; two files with unusual names, HYVY.EXE and A.EXE; and the execution of FTP.EXE. These are all 'Named' and highlighted (Figure 12).

Figure 12. Unusual File Names. (Lee, 2014, digital case files)

Looking within the timeframe, we see activities that suggest the adversary ran commands on the system (Figure 13). There is also another suspicious file named PE.EXE. These are all 'Named' and highlighted as well (Figure 13).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 13. Suspicious activity. (Lee, 2014, digital case files)

The keyword lists should contain the newly found keywords identified in the Prefetch output. As new ones get found, those will be added as well to help locate other indicators of compromise. The second artifact containing the execution of files is the ShimCache entries from the registry.

Running the keywords against the ShimCache artifacts produced one hit on file PE.EXE. Though the dates of the artifact don't show the time of execution, executables with the same write timestamp may be related. In this output, there is another executable, located in a strange location, which has the same timestamp as PE.EXE. Normally svchost.exe will reside in C:\Windows\System32. Its location in C:\Windows\System32\dllhost, makes it suspicious. Since the legitimate svchost.exe file shows up in a lot of logs, the keyword for this IOC will be '\dllhost\'. Both entries are 'Named' and highlighted (Figure 14).

Figure 14. Identical Timestamps. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Next, we'll look at the MFT file. After running our keywords against this file's output, we can now see the full paths of the suspicious artifacts (Figure 15).

Figure 15. Paths of suspicious files are displayed. (Lee, 2014, digital case files)

When looking at the keyword hits, you'll notice that they often appear in clusters. Looking at one of the clusters, we see a new artifact named SEKURLSA.DLL in the middle of several suspicious executables (Figure 16). Googling this file name shows us that it belongs to the Mimikatz tool mentioned previously. This discovery substantiates our guess that file hydrkatz.exe is Mimikatz in disguise. Therefore, we'll add Sekurlsa.dll to our keyword list.

Figure 16. Mimikatz .dll file. (Lee, 2014, digital case files)

Finally, it is easy to filter the data on a particular path to see if there are any other files of interest in the same directories. Looking at c:\windows\system32\dllhost\, we find the file WINCLIENT.REG, which we'll add to the keyword list (Figure 17).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

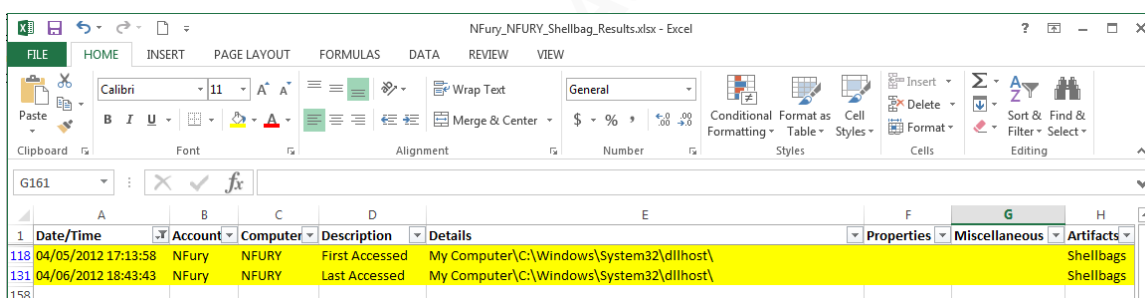*Figure 17. Winclient artifact. (Lee, 2014, digital case files)*

Using the keywords or IOCs discovered so far, we can start to look for other compromised systems on the network.

5.1.2. Win7-32-NROMANOFF (WKS-WIN732BITA)

Running our keyword list against workstation WKS-WIN732BITA (10.3.58.7) prefetch files generates hits on A.EXE and HYDRAKATX.EXE. Within the intrusion timeframe established earlier, we see the adversary possibly using native Windows tools to explore the host and network (Figure 18).

Figure 18. Possible native Windows tools used by the adversary. (Lee, 2014, digital case files)

Executing one after the other are two suspicious files with randomly generated names (Figure 19).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 19. Randomly generated file names. (Lee, 2014, digital case files)

Finally, there is a well-known artifact associated with the remote administrative command tool named PSEXEC by Microsoft. Administrators of the system could have used this tool to manage the system, but since it appears in our timeframe and may be an indication of lateral movement of the adversary, the keyword PSEXE will be added to our list to spot the 'Client' executable of PSEXEC.EXE and the 'Service' executable of PSEXESRV.EXE. Right below PSEXESRV.EXE is another artifact observed on workstation WKS-WINXP32BIT (but not noted), named SPINLOCK.EXE. Since this executable was run two minutes after PSEXEC and since the name of the file is not familiar to us, it will be added to the keyword list as well (Figure 20).

Figure 20. The keyword list has two more files added to it. (Lee, 2014, digital case files)
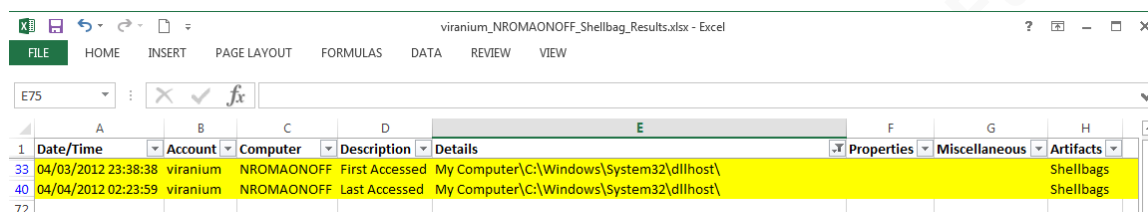
Keywords run against the Autoruns output gives us two hits. One on PSEXESRV.EXE and one for C:\Windows\System32\dllhost\svchost.exe (Figure 21).

Figure 21. Two keyword hits on Autorun entries. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
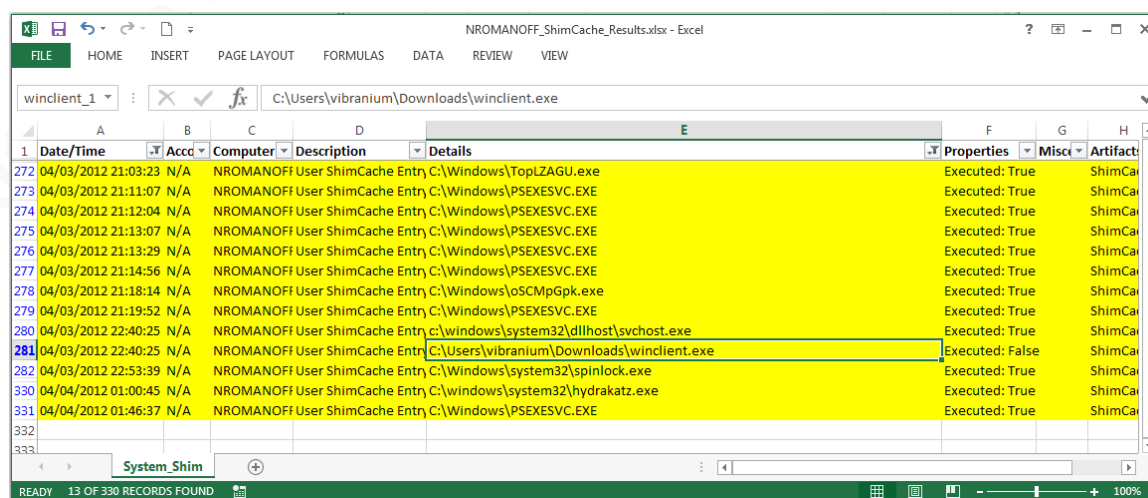
We find two additional hits when examining the Shellbag entries of user Vibranium (Figure 22).

Figure 22. Shellbag keyword hits. (Lee, 2014, digital case files)

Running the keywords against the ShimCache entries yields several interesting artifacts. First, we see that C:\Windows\Temp\a.exe is executed 47 times in one hour. Next, we find an entry with a file name from our keyword list, but with a different file extension. Recall that we saw a suspicious file called WINCLIENT.REG. The keyword list contained only the word WINCLIENT as the search term. This shortened keyword produced a hit on WINCLIENT.EXE which we found in user Vibranium's download folder (Figure 23). This user account is the same one seen above accessing the C:\Windows\System32\dllhost\ directory which contained the WINCLIENT.REG file from before.

Figure 23. Winclient keyword hit. (Lee, 2014, digital case files)

Last, every entry from 04/03/2012 onward is a hit, except three files. Looking at these three files, they appear suspicious and are flagged (Figure 24).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 24. Suspicious Entries found in the Time Frame. (Lee, 2014, digital case files)

Running the keywords against the MFT file shows us the paths of the artifacts of interest on this system (Figure 25).

Figure 25. File paths for keyword hits. (Lee, 2014, digital case files)

Running the keywords against the Application Event Viewer log produced two hits (Figure 26).

Figure 26. The keyword hits in the Application Event Viewer log. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

The Security Event Viewer log has several keywords in it (Figure 27).

Figure 27. The keyword hits in the Security Event Viewer log. (Lee, 2014, digital case files)

And finally, the System Event Viewer log has a bunch of hits (Figure 28).

Figure 28. The keyword hits in the System Event Viewer log. (Lee, 2014, digital case files)

Examining the Event Viewer logs using Event IDs instead of keywords, also yields some information that was not previously known. Pivoting on Event ID 5156 ("The Windows Filtering Platform has permitted a connection") shows an outbound connection from one of the suspicious executables to IP address 12.190.135.235 (Figure 29). This address will be added to the keyword list and possibly added to the company's network devices to log or block any connections still going to this location.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
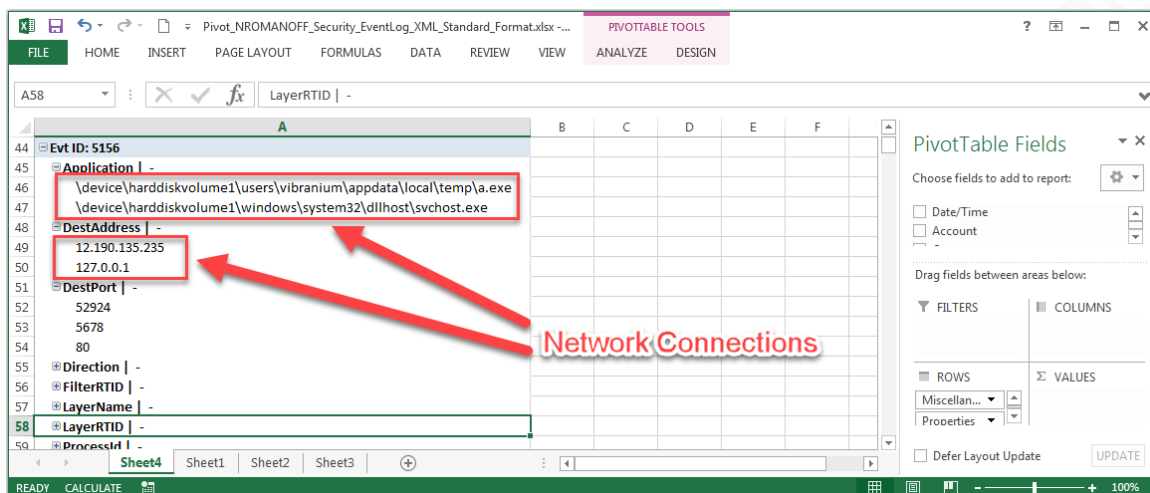
Figure 29. Outbound connection. (Lee, 2014, digital case files)

Examining the Event IDs in the CSV output of the Event Viewer Security log shows us some other activity that could be related to the intrusion. First, we filter on Event ID 4624 ("An account was successfully logged on") Type 10 ("RemoteInteractive"), which shows us possible lateral movement using the Remote Desktop protocol (Figure 30).
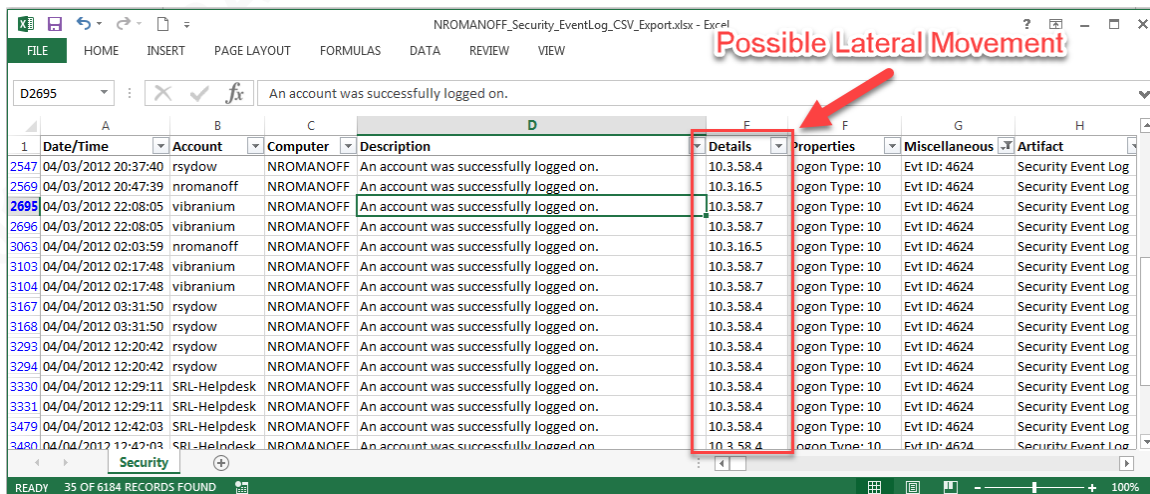
Figure 30. Possible lateral movement. (Lee, 2014, digital case files)

Next, we apply a filter on Event IDs 4717, 4724, 4732, 4733, and 4738. These Event IDs relate to changes made to user accounts. The results of the filter show activity by account RSYDOW against account SRL-Helpdesk (Figure 31).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 31. User account changes. (Lee, 2014, digital case files)

Finally, examining the System Event Viewer log by Event ID produces several more keywords. The Event ID filtered on is 7045 ("A service was installed in the system"). This filter shows us hits on several suspicious executables that we've already seen, but also provides us with unique Service Names of Mys, winsvchost, and MqlXmtLRaYQDMsvljY (Figure 32). We added these three new terms to the keyword list.
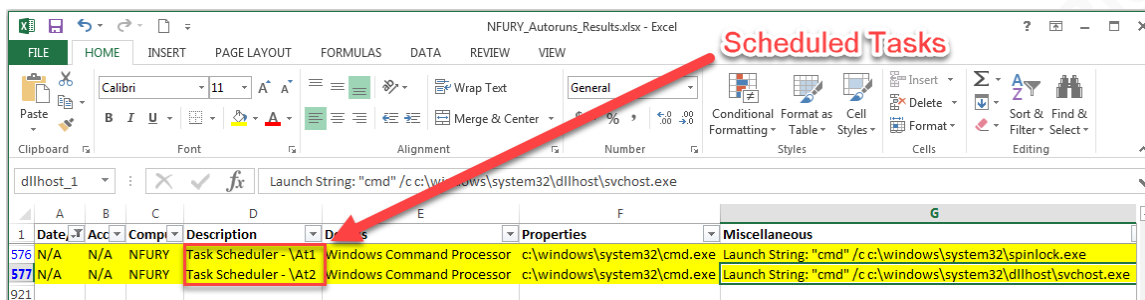
Figure 32. The keyword list has Service names added to it. (Lee, 2014, digital case files)

5.1.3. Win7-64-NFURRY (WKS-WIN764BITB)

Running our keyword list against workstation WKS-WIN764BITB (10.3.58.6) Autorun entries shows us Scheduled Tasks that run two of the suspicious executables that we have seen on other machines on this network (Figure 33).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 33. Suspicious scheduled tasks. (Lee, 2014, digital case files)

A search against NFurry's Shellbag entries shows the directory C:\Windows\System32\dllhost\ where several suspicious executables reside (Figure 34).
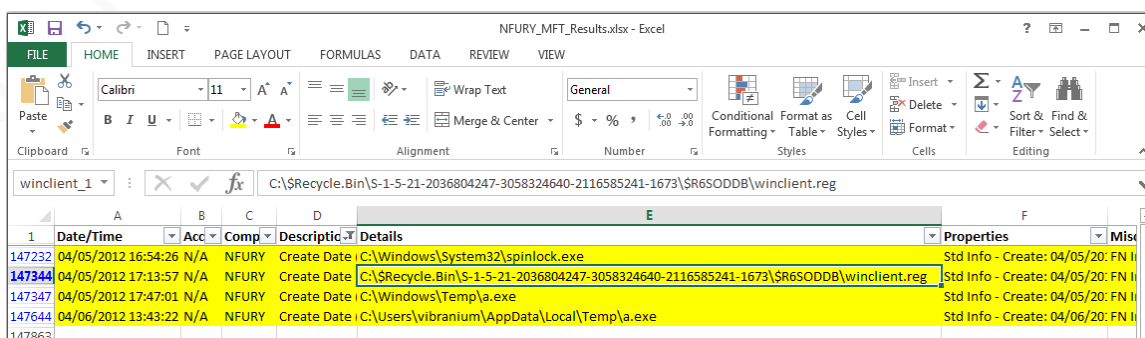
Figure 34. Shellbag keyword hits. (Lee, 2014, digital case files)

A keyword search against the MFT entries on the system shows the suspicious WINCLIENT.REG file in a user's Recycle Bin (Figure 35).

Figure 35. Winclient in user's recycle bin. (Lee, 2014, digital case files)

When we look at the Application Event Viewer log, two entries show up for the suspicious SVCHOST.EXE file (Figure 36).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 36. Keyword hits in the Application Event Viewer log. (Lee, 2014, digital case files)

And finally for this host, in the System Event Viewer log, several instances of PSEXEC and PSEXESVC are found running as services (Figure 37).

Figure 37. Keyword hits in the System Event Viewer log. (Lee, 2014, digital case files)

5.1.4. Win2008R2-Controller (CONTROLLER)

The last host (IP Address 10.3.58.4) examined in this case study is the domain controller. When we look at the Autorun entries, we discover a scheduled task to run SPINLOCK.EXE (Figure 38).

Figure 38. Suspicious scheduled task. (Lee, 2014, digital case files)

Examining the MFT log shows PSEXEC and Spinlock on the system (Figure 39).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 39. Keyword hits in MFT file. (Lee, 2014, digital case files)

The Application Event Viewer records the same SPINLOCK.EXE file in Windows Error Reporting (WER) (Figure 40).

Figure 40. Keyword hits in the Application Event Viewer log. (Lee, 2014, digital case files)

The Security Event Viewer log (Figure 41) and the System Event Viewer log (Figure 42), both show the PSEXESVC.EXE service running.

Figure 41. Keyword hits in the Security Event Viewer log. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 42. Keyword hits in the System Event Viewer log. (Lee, 2014, digital case files)

5.1.5. Master IOC Spreadsheet

Taking all the hits found from each of the workstations and combining them into one master spreadsheet allows the responder to see, among other benefits, what the adversary is doing chronologically; what artifacts they used across systems; and what capabilities they deployed. To demonstrate the techniques outlined in this paper, the standard output in the below images have been manipulated to fit the relevant information into the screenshots.

When we look at the timeline of events, the incident appears to have started on 04/03/2012 with the exploitation of host TGUNGAN (WKS-WINXP32BIT), then laterally moving to host NROMANOFF (WKS-WIN732BITA), then on to the Domain Controller (CONTROLLER) and finally to NFURY (WKS-WIN764BITB) (Figure 43).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 43. A timeline of events. (Lee, 2014, digital case files)

Tools used across hosts by the adversary include svchost.exe (Figure 44), a.exe (Figure 45), spinlock.exe (Figure 46), winclient (Figure 47), and psexesvc (Figure 48).

Figure 44. SVCHOST.EXE keyword hits. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 45. A.EXE keyword hits. (Lee, 2014, digital case files)

Figure 46. SPINLOCK.EXE keyword hits. (Lee, 2014, digital case files)

Figure 47. WINCLIENT keyword hits. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 48. PSEXESVC.EXE keyword hits. (Lee, 2014, digital case files)

The capabilities of the adversary are also significant to know to defend and respond to the intrusion. Figure 48 has already shown that the intruder is likely using PSEXEC from Microsoft to assist in moving laterally. There is also evidence that the adversary is running malware through Scheduled Tasks (Figure 49) and is using the tool Mimiatz tool to steal user credentials (Figure 50).

Figure 49. Scheduled Tasks. (Lee, 2014, digital case files)

Figure 50. Mimikatz artifacts. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

6. Conclusion

Using Microsoft Excel in your investigation during an incident can help bring lots of different data sources together into one location with a standard format for quick and easy analysis of the data. It also allows the analyst to manipulate the data in ways that can bring out indicators of compromise missed when buried in unrelated events contained in the original log files. This paper only touched on a few of Microsoft Excel's features and capabilities. The more you can automate the processes involved in analyzing the data, the better the tool becomes. To take it to a higher level, consider learning more advanced concepts in programming in VBA and taking advantage of add-ins/plugins that Microsoft and third party entities offer that can expand the analysis of your data even further.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

7. References

References

Carrier, B. (2005). *File System Forensic Analysis*. Addison-Wesley Professional

Carvey, H. (2014). *Windows Forensic Analysis Toolkit, 4th Edition*. Syngress

Caswell, B., Beale, J., & Baker, A. (2007). *Snort Intrusion Detection and Prevention Toolkit*. Syngress

Cowen, D. (2013). *Computer Forensic InfoSec Pro Guide*. McGraw-Hill

Define and use names in formulas - Excel. (n.d.). Retrieved from https://support.office.com/en-us/article/Define-and-use-names-in-formulas-4d0f13ac-53b7-422e-afd2-abd7ff379c64

Jelen, B., & Alexander, M. (2006). *Pivot Table Data Crunching*. Indianapolis, IN: Que Publishing

Lee, R. (2014). *SANS Forensic 508 Advanced Computer Forensic Analysis and Incident Response, Stark Research Labs Intrusion: Exercise Workbook*.

Microsoft Excel. (n.d.). In Wikipedia. Retrieved November 18, 2016, from https://en.wikipedia.org/wiki/Microsoft_Excel#Macro_programming

Nemeth, E., Snyder, G., Hein, T., & Whaley, B. (2010). *Unix and Linux System Administration Handbook, Fourth Edition*. Prentice Hill

Parisi, T. (2015, June 17). *Caching out: The value of shimcache for investigators « Threat Research Blog | FireEye Inc*. Retrieved from https://www.fireeye.com/blog/threat-research/2015/06/caching_out_the_val.html

Roesch, M. (2003). *Network intrusion detection system mode | NIDS mode output options*. Retrieved from http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node6.html

Russinovich, M., & Margosis, A. (2011). *Windows Sysinternals Administrator's Reference*. McGraw-Hill

Windows event log essentials - Windows event log FAQ. Basic explanation of Windows event logs. (n.d.). Retrieved from https://eventlogxp.com/essentials/windowseventlog.html

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

8. Appendix

8.1.   Appendix A - Open Source Tools

Here is a listing of the tools used in this document:

- NOTEPAD++, v7.3.2 (https://notepad-plus-plus.org/)

- Shellbags Explorer, v0.7.0.0 (http://binaryforay.blogspot.com/p/software.html)

- Bro, v2.5 (https://www.bro.org/)

- Snort, v2.9.9.0 (https://www.snort.org/)

- IPTables, v1.6.1 (https://git.netfilter.org/iptables/)

- Wevtutil.exe, v6.1.7600.16385 (Native Windows 7 tool)

- Autorunsc.exe, v13.7 (https://technet.microsoft.com/en-us/sysinternals/bb842062)

- Shimcacheparser.py, v1.0 (https://github.com/mandiant/ShimCacheParser)

- Pasco.exe, v1.0 (https://www.mcafee.com/us/downloads/free-tools/pasco.aspx)

- analyzeMFT, v2.0.18 (https://github.com/dkovar/analyzeMFT)

- parse_prefetch_info, v1.5 (http://redwolfcomputerforensics.com – dead link)

- Wireshark, v2.2.4 (https://www.wireshark.org/)

- CYGWIN, v2.7.0. (https://www.cygwin.com/)

    Within CYGWIN Environment:

    o   SED - https://www.gnu.org/software/sed/

    o   CUT – part of https://www.gnu.org/software/coreutils/coreutils.html

    o   CAT – part of https://www.gnu.org/software/coreutils/coreutils.html

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

8.2.   Appendix B – Resources

Here is a listing of books and blog web sites that I have found beneficial in learning Excel and VBA Programming:

- Excel 2013 Power Programming with VBA by John Walkenback

- Excel 2013 Bible by John Walkenback

- http://analysistabs.com/

- https://powerspreadsheets.com/

- http://stackoverflow.com/

- http://wellsr.com/

- https://www.ablebits.com/

- http://www.mrexcel.com/

- https://www.techonthenet.com/excel/index.php

- https://www.thespreadsheetguru.com

8.3.   Appendix C – Files Available on GitHub

I have provided code to automate much of the processes described in this document. If there are issues running the code, check to make sure you are using the same version of software listed in Appendix A.

The VBA code is separated up into two categories. One category contains individual code for each of the log files mentioned in this document along with a few miscellaneous scripts for specific tasks. The other category contains two 'Master' spreadsheets that comprise many of the individual scripts into one document for ease of execution. You can find the code at https://github.com/gregory-lalla/GCIH_Gold under the 'Code' directory. Each category has two sub-categories. The sub-categories are explained below.

Module Files:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

The files under this sub-category are individual modules that can be added to macro-enabled spreadsheets (see Section 3.5 for directions on adding macros to a spreadsheet). You can find the code at GitHub site under the Code/Singles/Module_BAS_Files directory. Each log file has code designed specifically for the output shown in this paper. There are four types of modules. The first type will search the data for specific keywords and will have the word 'keyword' in the file name. The second type will search the data for specific Event IDs and will have the word 'EventID' in the file name. The third type will format the data per the instructions provide in this paper and will contain the words 'Standard_Format' in the file name. The last type will be code which is designed to perform a specific task and will not have any of the words mentioned above in the file name. The following is a list of the task specific macros and a description of each:

- Binary_Hex2Ascii_Conversion_Module:

  This script will convert a column of hexadecimal encoded characters to human readable ASCII encoding. This script will work on an XML export of Microsoft Event Viewer data.

- Insert_Headers_Module:

  This script will insert the standard headers described in this paper to the spreadsheets first row.

- EventLogs_Application_CSV_Unique_IDs_per_Sheet_Module:

  This script will take each unique Event ID and create an individual spreadsheet (tab) containing only those Event IDs. This script will work on a CSV export of Microsoft Event Viewer Application log data.

- EventLogs_Security_CSV_Unique_IDs_per_Sheet_Module:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

This script will take each unique Event ID and create an individual spreadsheet (tab) containing only those Event IDs. This script will work on a CSV export of Microsoft Event Viewer Security log data.

- EventLogs_System_CSV_Unique_IDs_per_Sheet_Module:

    This script will take each unique Event ID and create an individual spreadsheet (tab) containing only those Event IDs. This script will work on a CSV export of Microsoft Event Viewer System log data.

Macro Files:

The files under this sub-category also contain individual scripts designed specifically for each log file. The scripts here, however, are contained within a Macro-enabled Excel file. There is a button on the first spreadsheet that will prompt the user to specify the log file they wish to process. The script will then run and produce a unique output file name based on the hostname that produced the log. These Excel files contain the same types of scripts mentioned above, except there are no task-specific modules for hexadecimal to ASCII conversion or inserting of headers. You can find the code at the GitHub site under the Code/Singles/Excel_Macro_Files directory.

Singles Combined:

The file under this sub-category is named 'Master-Single.xlsm' and combines all the code from the Macro Files sub-category. Each log file has its own button that when pushed will run the same code as the Macro Files sub-category. This file is available on the GitHub site at https://github.com/gregory-lalla/GCIH_Gold/blob/master/Code/Combined/Singles_Combined/Master-Single.xlsm.

Master IOC Combined:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

The file under this sub-category is named 'Master-IOC.xlsm' and has two differences from the 'Master-Single.xlsm' document. First, the result of the script not only writes to a unique file, but is appended to an overall Master IOC Excel file. When opening the document, the user is prompted to either select a previously created Master IOC file or to create a new one. This file is then used to hold the results of all the scripts run. The other difference is that instead of a standard format button, there is a button to filter the results of each log file by date to remove data that is outside the timeframe of an incident. This file is available on the GitHub site at https://github.com/gregory-lalla/GCIH_Gold/blob/master/Code/Combined/Master_IOC_Combined/Master-IOC.xlsm.

There is also a documents directory located at https://github.com/gregory-lalla/GCIH_Gold/blob/master/Docs with several subdirectories. The 'Supplement' subdirectory contains a document named 'Additional_Log_Formatting_Instructions.docx' which covers instructions for formatting the logs mentioned in Section 4.2 of this paper. It also contains a file named 'Pivote_Table_Example.docx' which shows examples of the content discussed in Section 3.6 of this paper. Last, there is a file named 'Complete_GIAC_GCIH_Gold_Paper_Greg_Lalla.docx' which is the original document before it was edited to fit the requirements of a GIAC Gold Paper (Because of file size limits on Github, the document has been compressed and split into three files. You can recombine the compressed files with 7-zip). The 'Template' subdirectory contains an Excel Macro-Enabled Template named 'standard_format.xltm' you can use when manipulating data from log files not discussed in this paper. The last subdirectory in the 'Docs' folder is 'Worksheet_Functions.' This folder contains a document named 'Worksheet_Functions_in_Paper.docx' which contains the worksheet functions used in this paper, along with a description of what each one accomplishes.

The last directory at the root of the GitHub page is named 'Misc' and contains the SED script file named 'months.sed,' which fixes the Date/Time fields of Linux system

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

logs. You can find the SED script at https://github.com/gregory-lalla/GCIH_Gold/blob/master/Misc/SED_Month_Replacement_Filter/months.sed.

8.4.   Appendix D: Windows Logs Explained

For brevity, anything discussed previously will be not be explained again but only referenced.

8.4.1. Event Viewer Logs

Because Event Viewer logs can contain hundreds of thousands of entries, it is essential to reduce the data to a manageable level before importing it into Excel. One way to do this is to use XML filtering in Event Viewer Custom Views which is more efficient than filtering through the GUI. There are many types of expressions you can use, but two of the more useful are sorting by timeframe and sorting by a username (See Figure 51 and 52). For more information about XML filtering, see the article at https://blogs.technet.microsoft.com/askds/2011/09/26/advanced-xml-filtering-in-the-windows-event-viewer/.
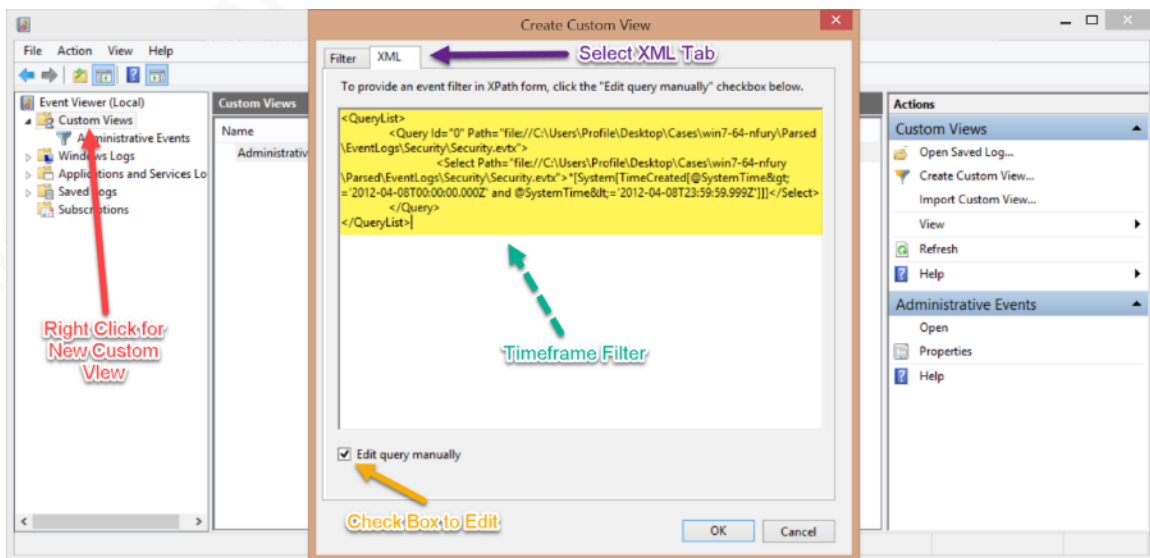
Figure 51. Event Viewer XML Timeframe Filter. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
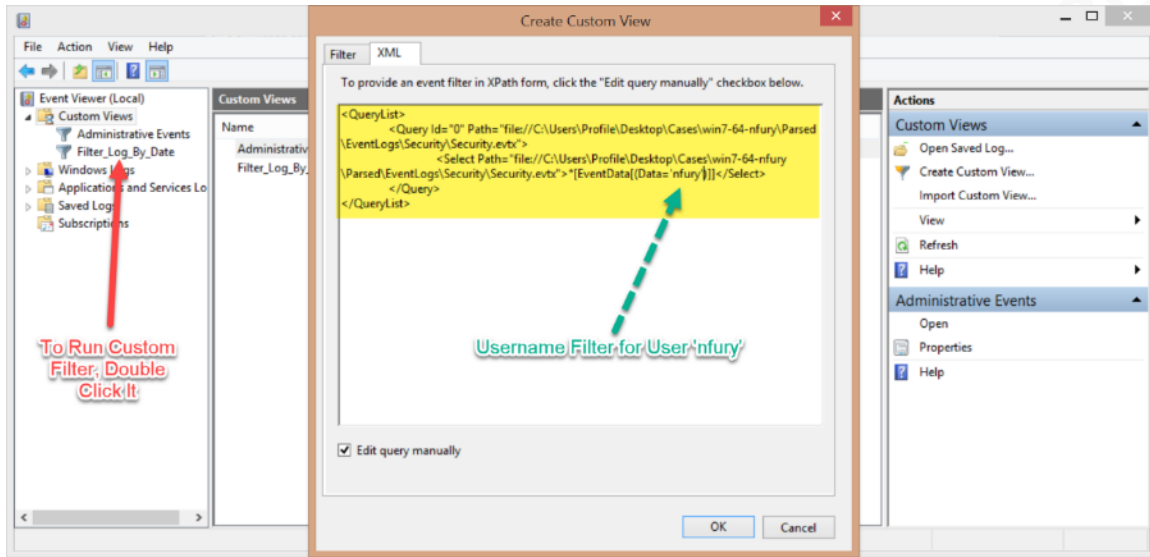
Figure 52. Event Viewer XML Username Filter. (Lee, 2014, digital case files)

An even easier method, avoiding the slow GUI, is to use the native command line utility called WEVTUTIL.EXE to export the logs. This tool comes with its challenges such as producing XML files that Excel cannot open. When running the command, there are several options to massage the data to a format that Excel accepts. To get the correct results, you may need to mix and match the switches until you get an XML file that imports into Excel. Here is an example of a command that you can run which filters the data based on a timeframe:

wevtutil qe "<path_to_system.evtx>" /lf:true /q:"*[System[TimeCreated[@SystemTime>='2015-07-10T00:00:00.000Z' and @SystemTime<='2015-07-13T23:59:59.999Z']]]" /f:RenderedXML /e:root > System.xml

A final way to get the output you want from Event Viewer logs is to use Powershell, which is a topic beyond this paper, but well worth the effort to try and learn.

Once you have reduced the data, you should then export it from the utility. Event Viewer offers four different formats to export the data: .EVTX, .TXT, .CSV, and .XML (See Figure 33). This paper will look at the .CSV format and the .XML format. Neither of these formats includes all the information from the Event Viewer logs, and the data may need to be combined to get a full picture of what occurred during each event.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
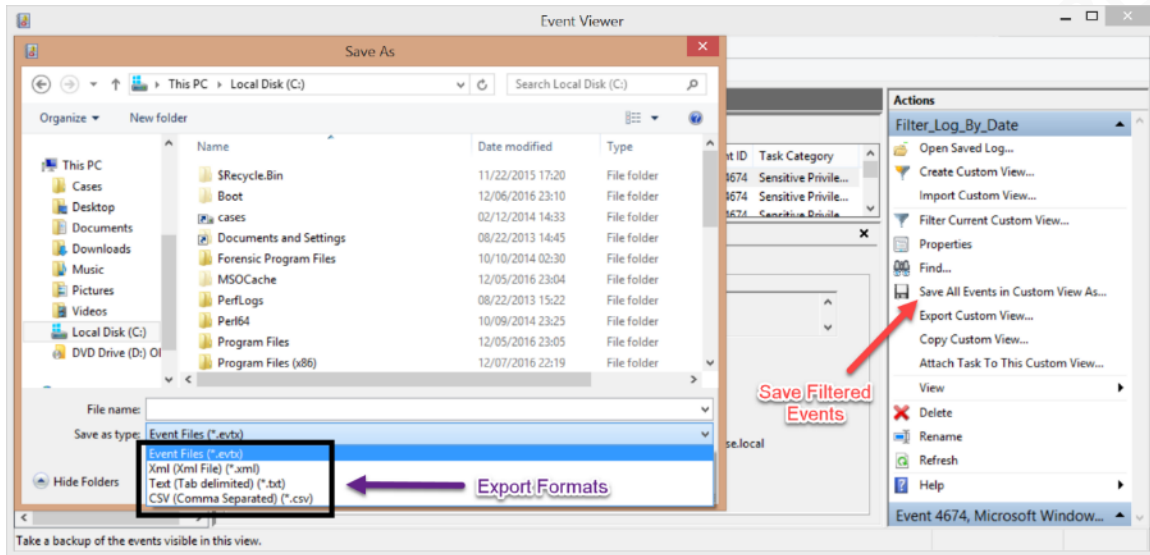
Figure 53. Save Filtered Event Viewer logs. (Lee, 2014, digital case files)

In this section, two event logs are shown, each with a slightly different output. With these two examples, you can manipulate other Event Viewer logs in a similar manner. First, the Security.evtx log, when exported as a .CSV file produces cells with newline and carriage returns, which makes the data unwieldy. You will remove the carriage return character (ASCII Code 13), and replace the newline character (ASCII Code 10) with the '#' character which will be used as a delimiter to split the data into columns. You can automate this task with the following Excel Function:

=SUBSTITUTE(SUBSTITUTE(F2,CHAR(13),""),CHAR(10),"#")

Place the above function in an empty cell adjacent to the cell that needs to have the newline and carriage return characters replaced (Figure 54). The blank cell will then contain the new contents as its value, but will also contain the formula used in the function.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
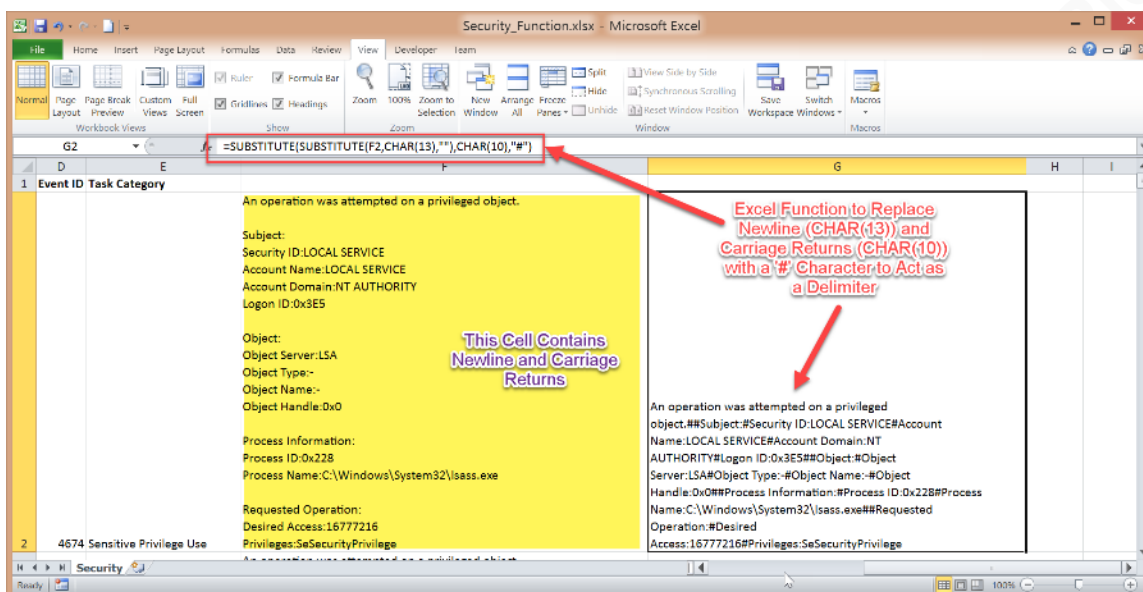
Figure 54. Remove Newline and Carriage Return Characters from Cell Contents. (Lee, 2014, digital case files)

To produce a similar result in every cell in the column, highlight the cell with the function and double-click on the bottom right-hand corner of the cell. This action will copy the function to every cell in the column. The arguments used by the function will be updated to reflect the correct cell locations (Figure 55).
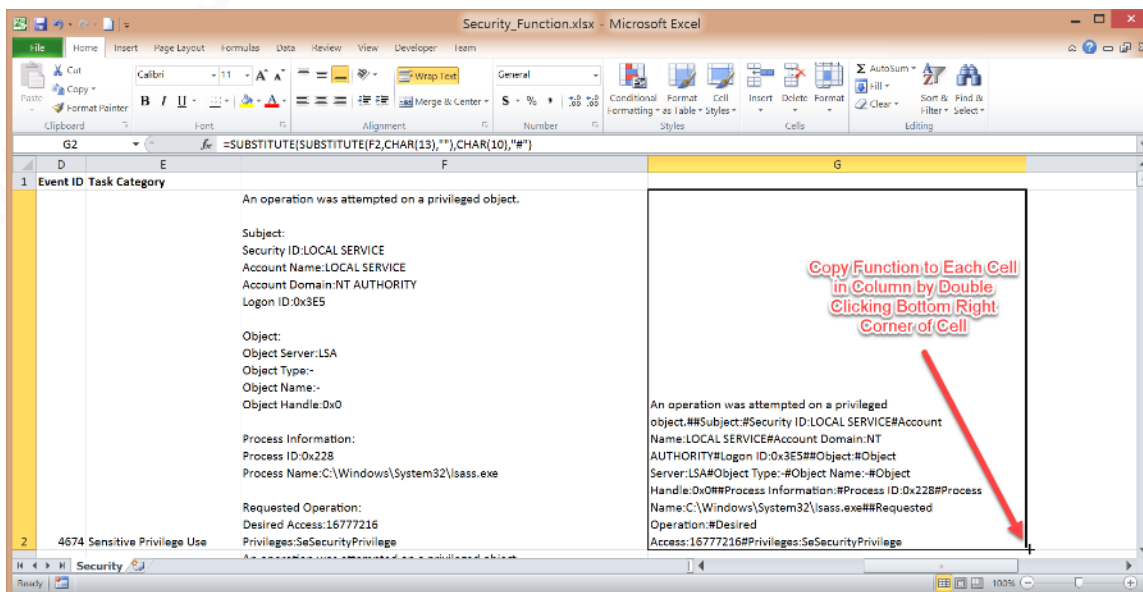
Figure 55. Copy Function to All Cells in Column. (Lee, 2014, digital case files)

To prevent the function from being accidently changed and to make it easier to manipulate the contents of the cell, the values within the new cells will be copied and

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

pasted over the formula. To perform this action, select the entire column that contains the functions, right click the highlighted cells, select 'Copy,' right click the highlighted cells again and select the 'Paste Value' icon which has the numbers 123 in the image (Figure 56).
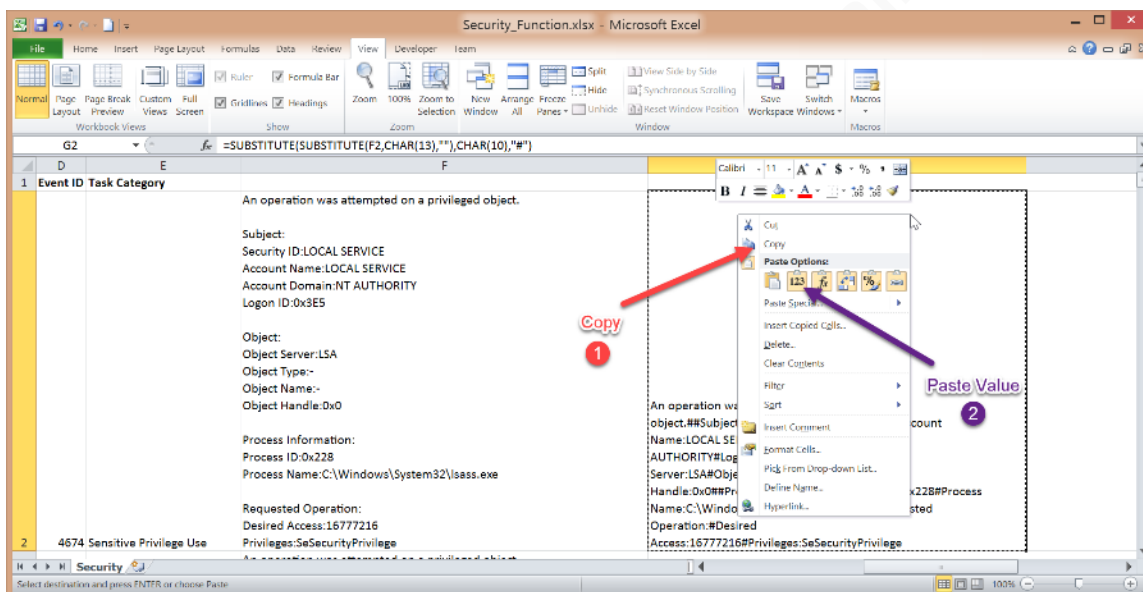
Figure 56. Copy and Paste Value of cell over the Function in the cell. (Lee, 2014, digital case files)

With the data copied to a new column in the correct format, you can delete the original column. Since the new column has a delimiter of '#' in the data, the 'Text to Column' feature on the 'Data' ribbon can be used to separate the data into multiple columns (Figure 57).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
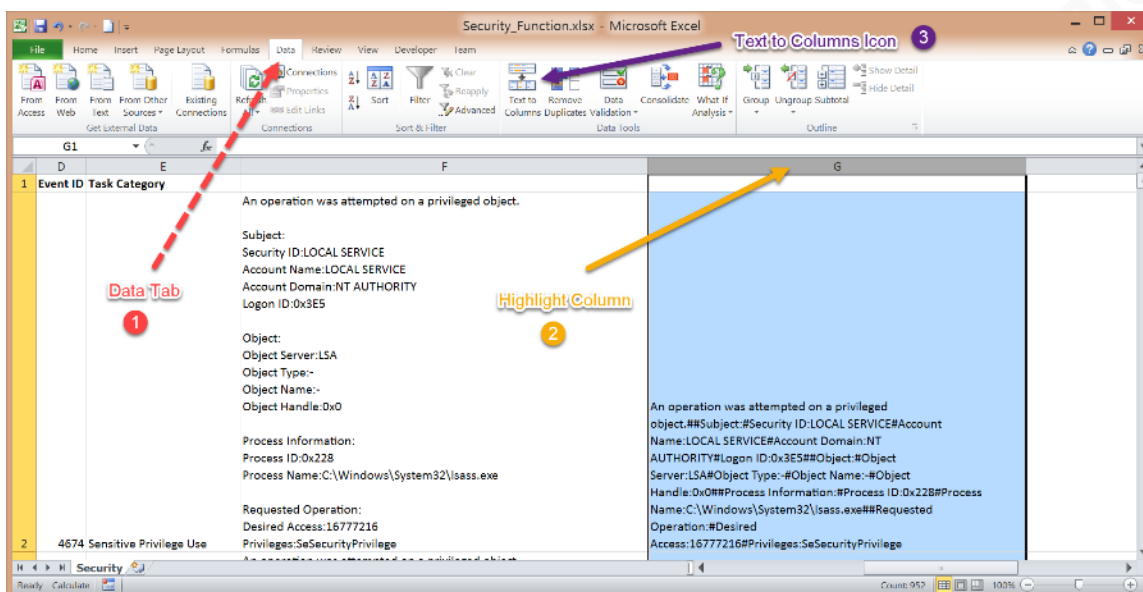
Figure 57. Text to Columns. (Lee, 2014, digital case files)

On the 'Text to Columns' wizard, select the defaults on Step 1 and 3. On 'Convert Text to Columns Wizard – Step 2 of 3' only check 'Other:' under 'Delimiters.' In the box next to 'Other:' type the '#' character. Also, make sure the 'Treat consecutive delimiters as one' checkbox is unchecked to keep data of a similar nature in the same columns (Figure 58).
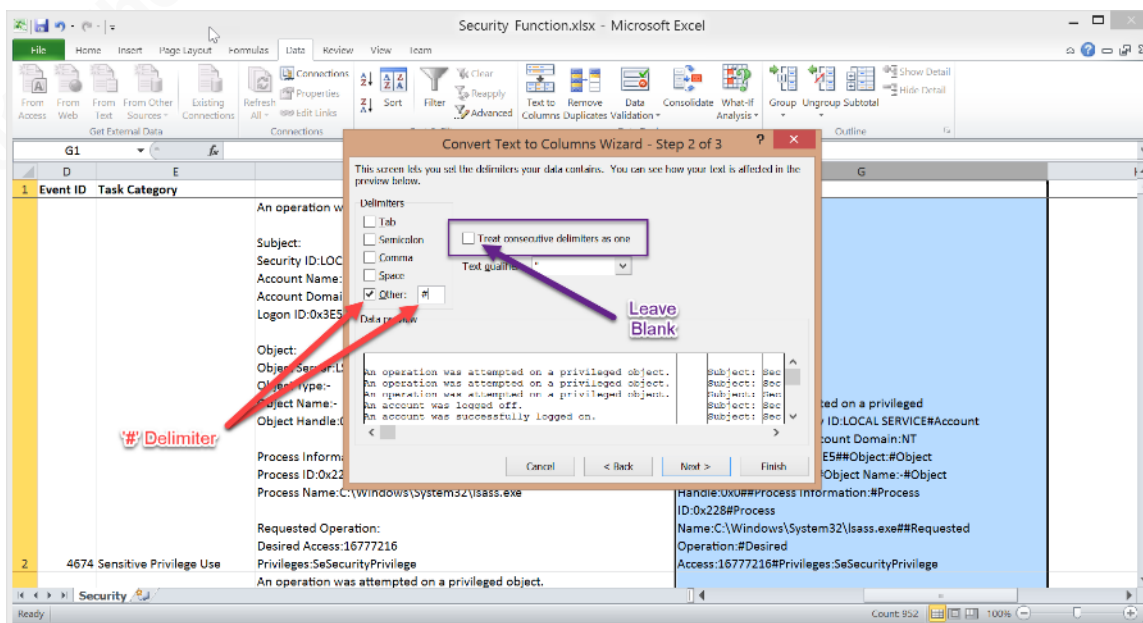
Figure 58. Text to Columns Delimited (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

The result should look like Figure 59 after 'Wrap Text' has been removed from the worksheet cells.

Figure 59. Text to Columns Result. (Lee, 2014, digital case files)

These actions should line up most of the data correctly, but you will find there are some columns with extra data or missing data. Here you'll need to do some manual work to get the data correctly aligned. First, run the Standard Format macro to better view the data. You need to keep in mind the data you want to retain, relative the Column Header names. Of the five headers already present, keep the 'Date and Time'; 'Source'; and 'Event ID' columns. The other two columns can be deleted (Figure 60).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 60. Keep the 'Date and Time' column and 'Event ID' column. (Lee, 2014, digital case files)

Change the 'Date and Time' column header to 'Date/Time,' the 'Source' header to 'Artifact' and the 'Event ID' header to 'Properties.' To understand what the value of the cells under 'Properties' represent, prefix each cell with the string 'Event ID: ' (Figure 61). You can do this by using the following function:

="Event ID: "&B2

Figure 61. Prefix cell value with a string. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Follow the steps outlined previously (the substitute function instructions near Figure 54) to get the desired results as shown in Figure 62.

Figure 62. Date/Time and Properties columns. (Lee, 2014, digital case files)

Using the filter dropdown arrows in each column header cell, delete columns that clearly don't contain information of value. Next, find the columns that correspond to the remaining Column Headers and label them accordingly. Finally, delete the columns that are not labeled. Remember, to make this process immensely easier, reduce as much data first by getting rid of rows based off the timeframe, keywords, and Event IDs you know will not contain information related to the intrusion. The final output would look like Figure 63 after putting the columns in the correct order.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 63. The final output of the CSV export data. (Lee, 2014, digital case files)

The second type of export from an Event Viewer log that we will look at is the XML format. This output is a lot easier to examine since there are already tags defining each column. As before, you export Event Viewer logs in the XML format by selecting 'Save All Events As…' or 'Save Filtered Log File As…' and choosing XML in the 'Save as type' dropdown list (Figure 53). Import the resulting .XML file into Microsoft Excel. You will get a popup box asking how to import the data. Select 'As an XML Table' and click on the 'OK' button (Figure 64).
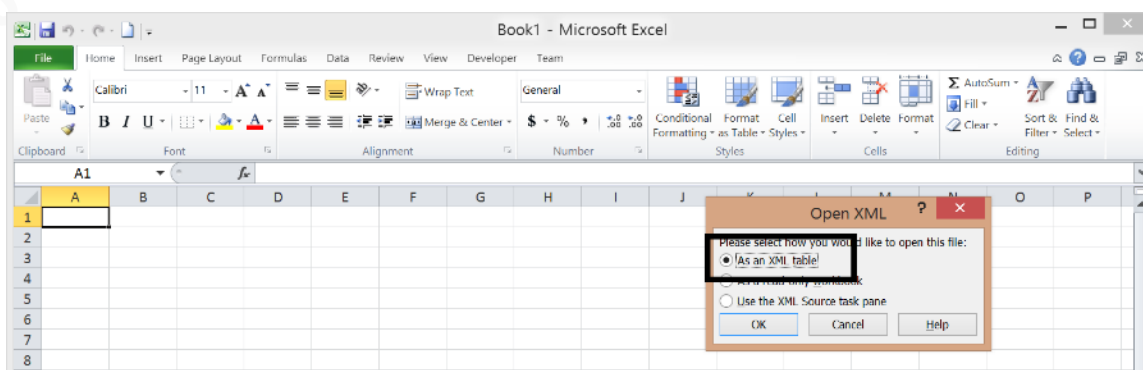
Figure 64. Open .xml file as an XML Table. (Lee, 2014, digital case files)

Once you have imported the data, it will look like Figure 65.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 65. Imported XML file as a Table. (Lee, 2014, digital case files)

One issue is the Date/Time format, which separates the date and time with the letter 'T' and includes milliseconds followed by the letter 'Z.' A function will be used to get the correct format of 'mm/dd/yyyy hh:mm:ss'. To make manipulating the data easier, you will convert the table to a normal range. This conversion can be done by selecting the 'Convert to Range' button on the 'Table Tools' ribbon and selecting 'Yes' on the popup window (Figure 66).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
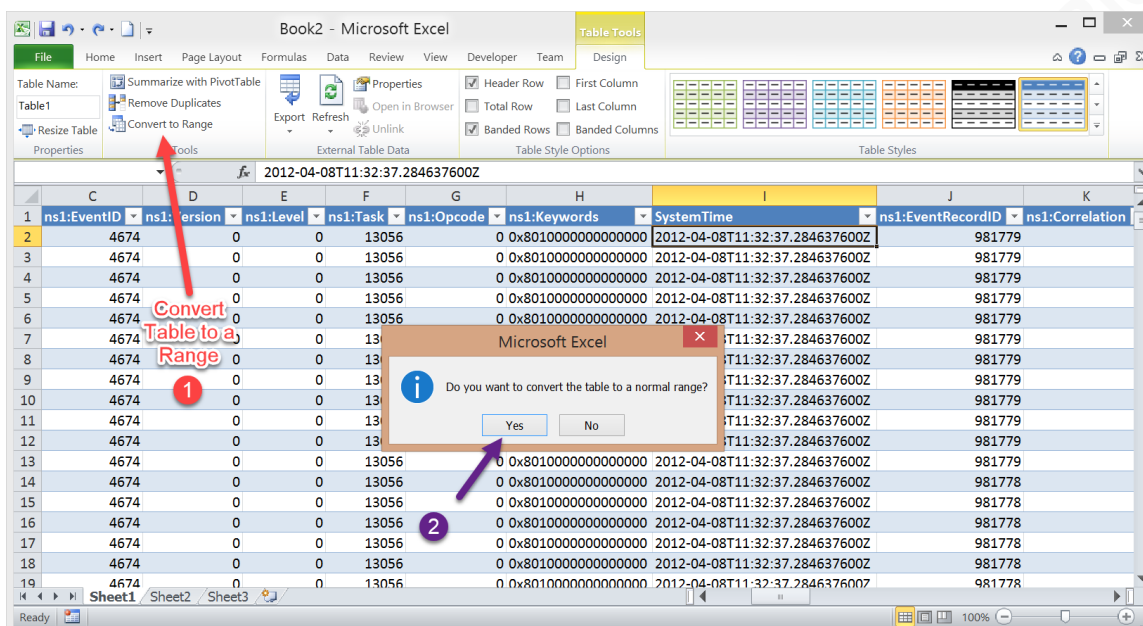
Figure 66. Convert Table to a Range. (Lee, 2014, digital case files)

In order prevent confusion from the highlighting of artifacts related to the incident, you should remove the formatting left over from the table layout (Figure 67).
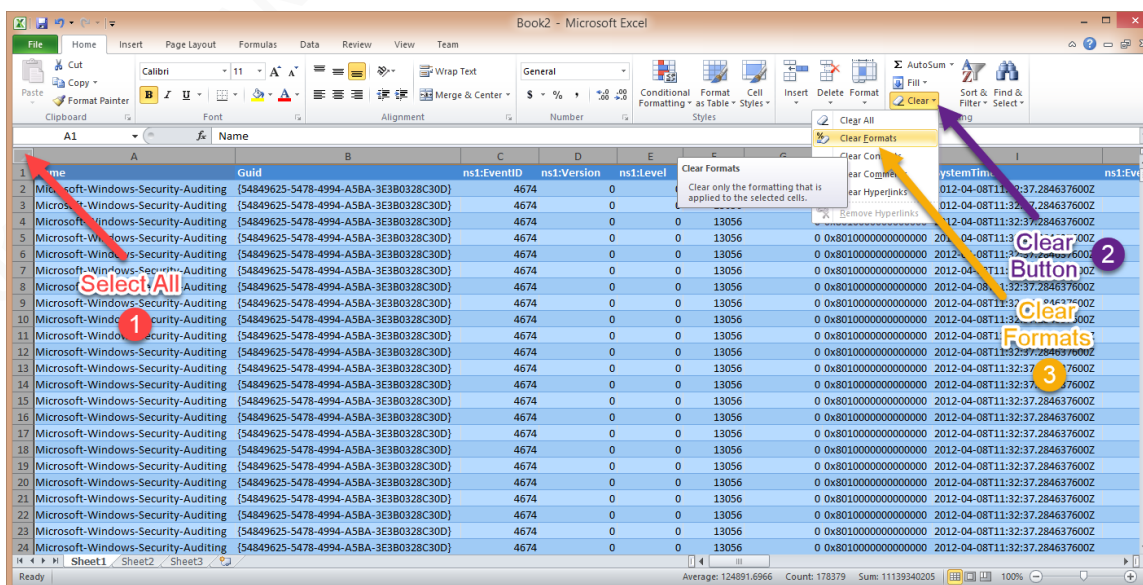
Figure 67. Clear Table Formatting. (Lee, 2014, digital case files)

To fix the Date/Time formatting, you can use the following function (Figure 68):

=0+SUBSTITUTE(SUBSTITUTE((MID(I2,1,19)),"T"," "),"-","/")

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
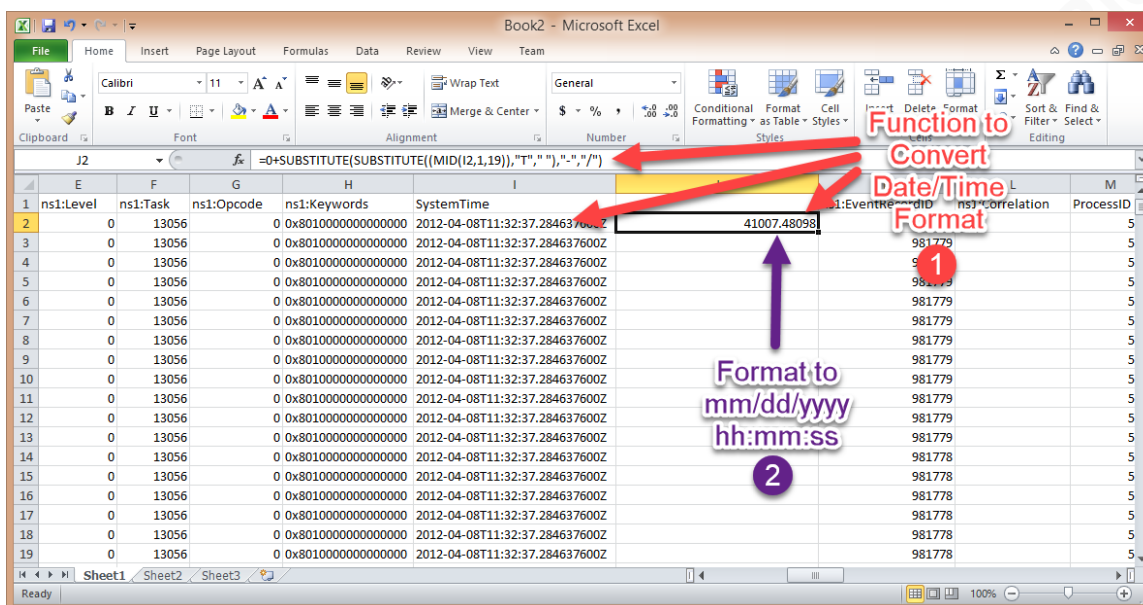
Figure 68. Function for Date/Time Formatting. (Lee, 2014, digital case files)

This function will display the Date/Time in number format. Apply the 'mm/dd/yyyy hh:mm:ss' format to the column for proper display.

Another issue you may run into is that some of the data is hex encoded (Figure 69). To convert the hex data to human readable ASCII data, you can run the 'Binary_Hex2Ascii_Conversion_Module.bas' code (https://github.com/gregory-lalla/GCIH_Gold/blob/master/Code/Singles/Module_BAS_Files/Misc/Binary_Hex2Ascii _Conversion_Module.bas) (Figure 70).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
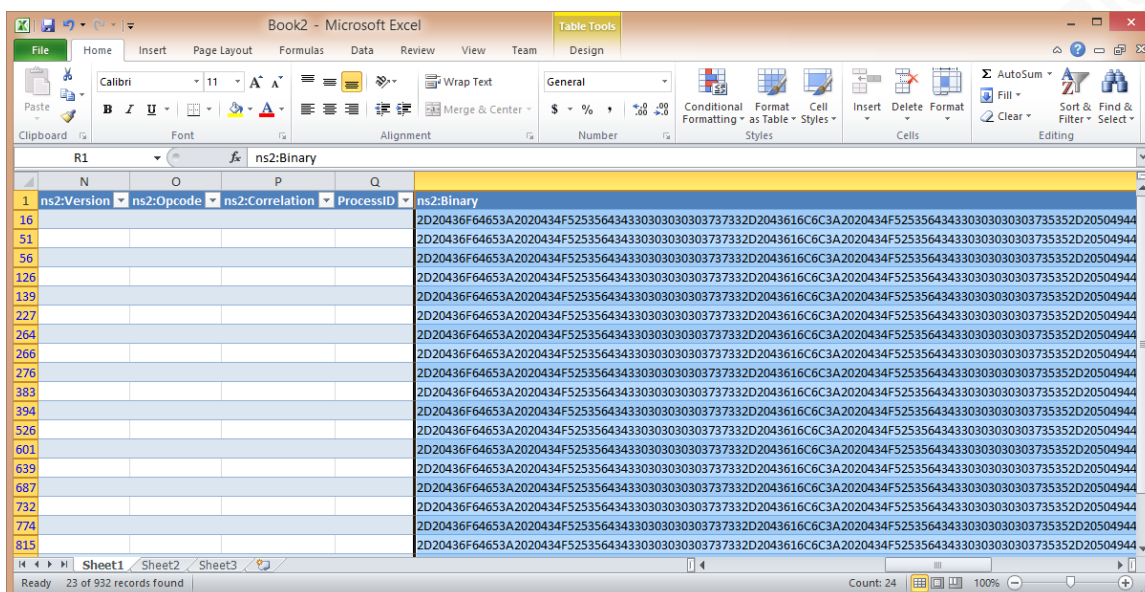
Figure 69. Hex Encoded Data. (Lee, 2014, digital case files)
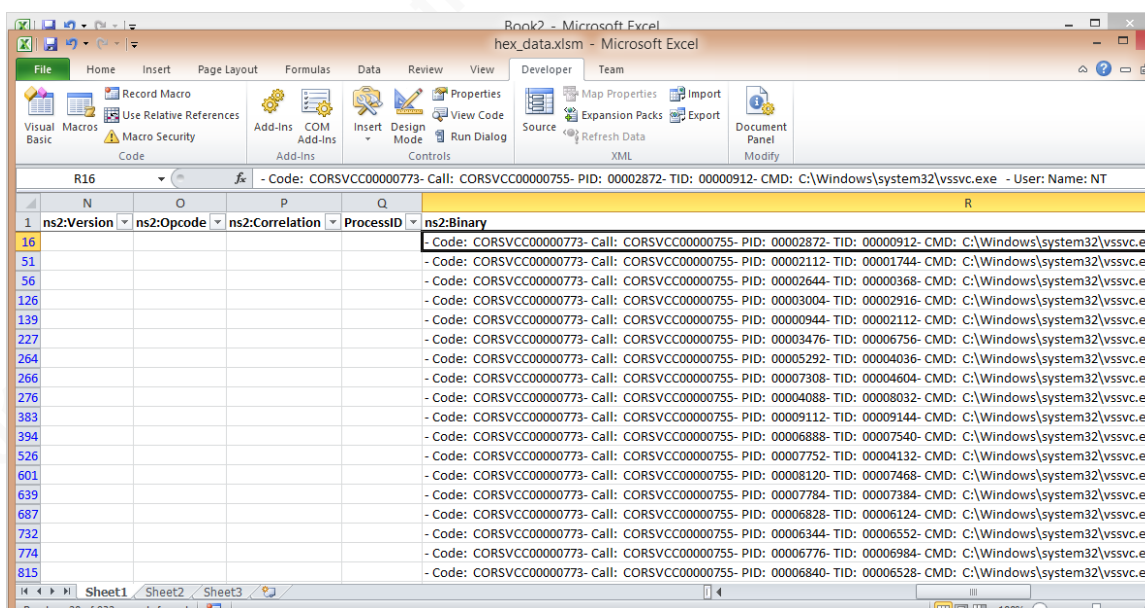
Figure 70. Hex Data Converted to ASCII Data. (Lee, 2014, digital case files)

Once you have fixed all the data conversion issues, it is a simple matter of finding the columns to keep, applying the headers, formatting the spreadsheet and putting the columns in the correct order (Figure 71). Once complete, you can begin to search for artifacts of interest.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 71. Final Formatted Spreadsheet from XML Data. (Lee, 2014, digital case files)

One last note on XML exported data. Each row does not correspond with only one Event Viewer log entry. There are multiple rows for each Event Viewer log entry identified by an 'EventRecordID' as shown in Figure 72. Therefore, when there is a hit on a keyword search, don't just copy over that one row to the master IOC file, but all the rows that have the same 'EventRecordID' number.

Figure 72. Event Record IDs in XML Exported Data. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

8.4.2. ShimCache Entries

We produced the ShimCache output used in this example by running the ShimCacheParser python tool against a Windows 7 SYSTEM registry file:

ShimCacheParser.py -i system -o System_Shim.csv

Open the .CSV output file in Excel and then save it as an Excel macro-enabled workbook file with an .XLSM file extension. Apply the Standard Format macro for a consistent look. (Figure 73).

Figure 73. Parsed ShimCache Data Standard Look. (Lee, 2014, digital case files)

In this data, there are a few things to note. First, since the date and time represent the Last Modified date of the executable, in the 'Miscellaneous' column it will be noted as such so that there is less confusion when looking at the final data in a timeline of events. Second, the 'Exec Flag' column needs to express that statement to understand what the 'TRUE' and 'FALSE' entries represent. Third, since there are not enough columns to match the headers we want, it is OK to leave the cells blank under those extra headers. Finally, we need to insert a column for the name of the computer. The final view of the ShimCache Excel spreadsheet should look like Figure 74.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 74. ShimCache Final Formatting. (Lee, 2014, digital case files)

8.4.3. Shellbags

We produced the Shellbags output used in this example by running the Shellbags Explorer tool against a Windows 7 USRCLASS.DAT registry file:

Sbecmd.exe --timezone="UTC" –d <path to directory containing usrclass.dat>

The output of the above command produces a tab delimited file with a .TSV extension. Import the file into Excel, choose TAB as the delimiter in the 'Text Import Wizard' and save it as an .XLSM file to run Macro's against the data (Figure 75).

Figure 75. Shellbags .TSV file imported into Excel. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Apply the Standard Format macro and delete the following columns in this order: R, O, N, M, J, I, H, G, F, D, C, B, A. You'll notice that the dates include a UTC offset value of '+00:00' (Figure 76). This value will need to be removed to have the Date/Time column properly formatted. The removal of the offset can be done with a simple search and replace.

Figure 76. Dates contain a UTC Offset. (Lee, 2014, digital case files)

After removing the offset and formatting the date and times correctly, the result should look like Figure 77.

Figure 77. Standard Format with Dates and Times fixed. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

There are six columns of Dates and Times. For this paper, the information sought is the first date the resource was accessed and the last date the resource was accessed. Manually going through all the entries and getting those two dates for each resource would consume too much time. To make finding the two dates and sorting the Date/Time column easier, we have provided VBA code on the GitHub site called 'Shellbags_Standard_Format_Module.bas' (https://github.com/gregory-lalla/GCIH_Gold/blob/master/Code/Singles/Module_BAS_Files/Shellbags/Shellbags_Standard_Format_Module.bas) which will automatically do the work for you. The script will produce two rows for each entry. One for the first time it was accessed and one for the last time it was accessed. If there is only one date for the entry or the dates shown are the same or within 3 seconds of each other, then only one row will be produced for that entry. Figure 78 shows the result of running the VBA code.

Figure 78. Shellbags View after running Shellbags_Date_Sorter code. (Lee, 2014, digital case files)

8.4.4. AutoRun Entries

To parse autorun data, we run the autorunsc.exe command against the registry files of a system. This parsing can be done offline by mounting an image of the system and pointing the tool to the mapped drive:

Autorunsc.exe -a * -c -m -s -z M:\Windows M:\Users\<profilename> >

autoruns_profilename.csv

The options in the above command are:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

-a *:   Check All
-c: CVS output - need to direct to a file
-m: Hide Microsoft Verified Signed
-s: Check signature
-z: offline analysis

Open the resulting .CSV file in Excel and apply the standard formatting (Figure 79).

Figure 79. Autoruns Standard Formatting. (Lee, 2014, digital case files)

You should delete the following columns: Enabled, Category, Publisher, MD5, SHA-1, and SHA-256. Two columns, 'Entry Location' and 'Entry' can be combined into one column to preserve data that might not fit into the suggested columns used in this paper (Figure 80).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
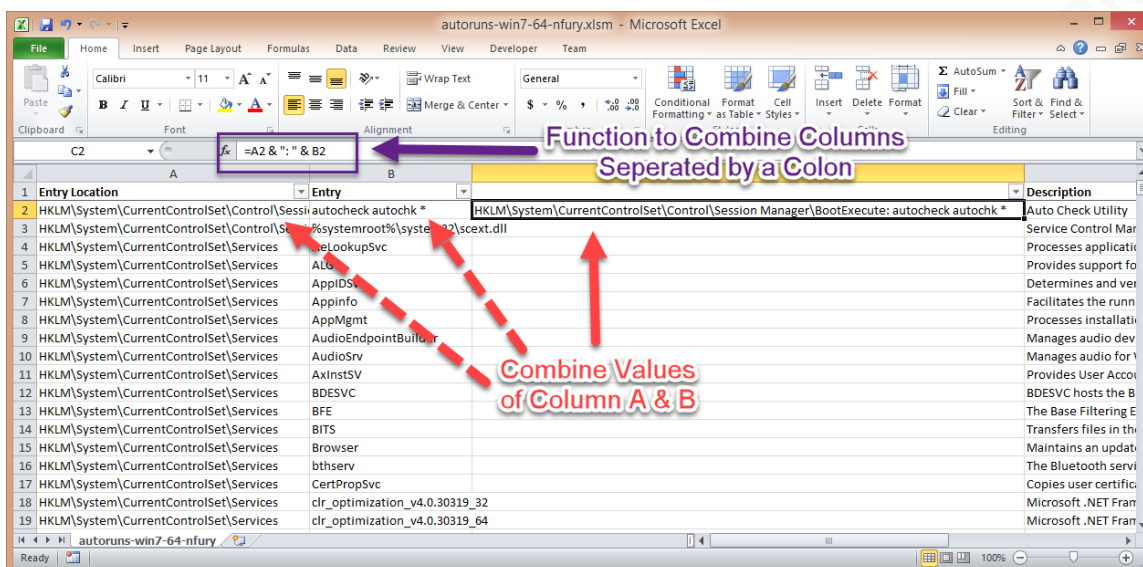
Figure 80. Combine Entry Location and Entry. (Lee, 2014, digital case files)

The combined columns should look like Column A in Figure 81.
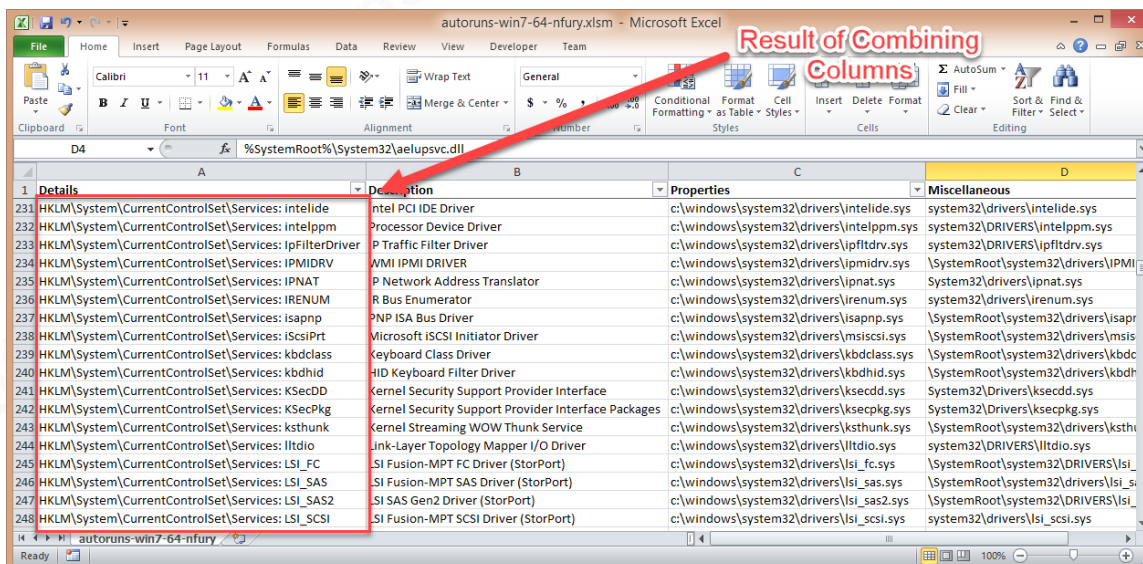
Figure 81. The result of Combining Columns. (Lee, 2014, digital case files)

8.4.5. Web Browser Logs

The example in this section will use the tool PASCO to parse the Internet Explorer files:

pasco.exe <path>

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

After you run the above command, open the resulting TSV file in Excel using tab as the delimiter. You should manually delete the first two rows which are not needed (Figure 82).
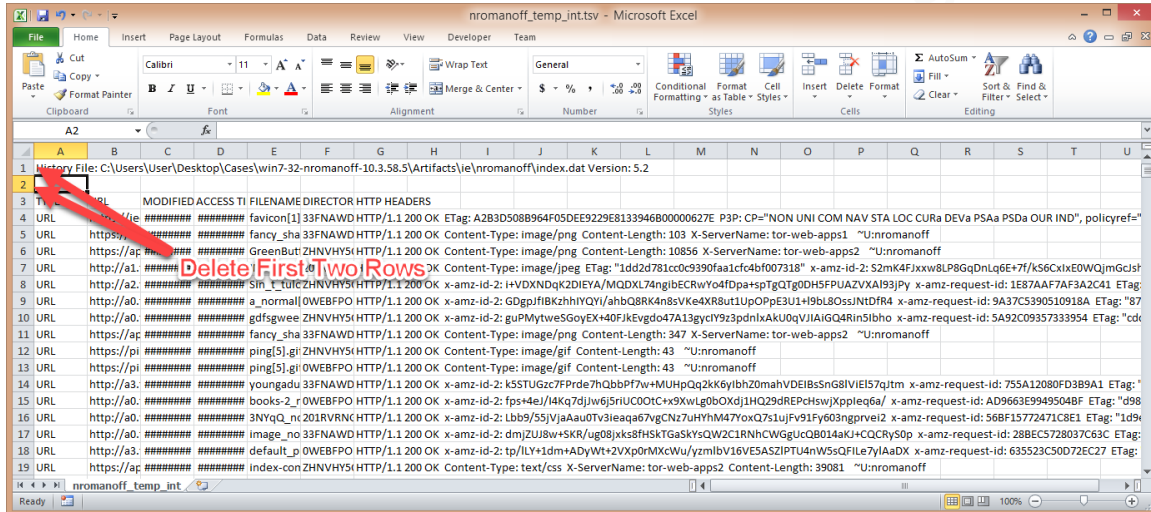
Figure 82. Delete first two rows. (Lee, 2014, digital case files)

You should delete the following columns: Modified Time, and Directory. Change the 'ACCESS TIME' column formatting to 'mm/dd/yyyy hh:mm:ss' and move that column to Column 'A.' Apply the standard formatting and headers and then manually fill in the missing values in the empty columns. Once complete, it should appear like Figure 83.
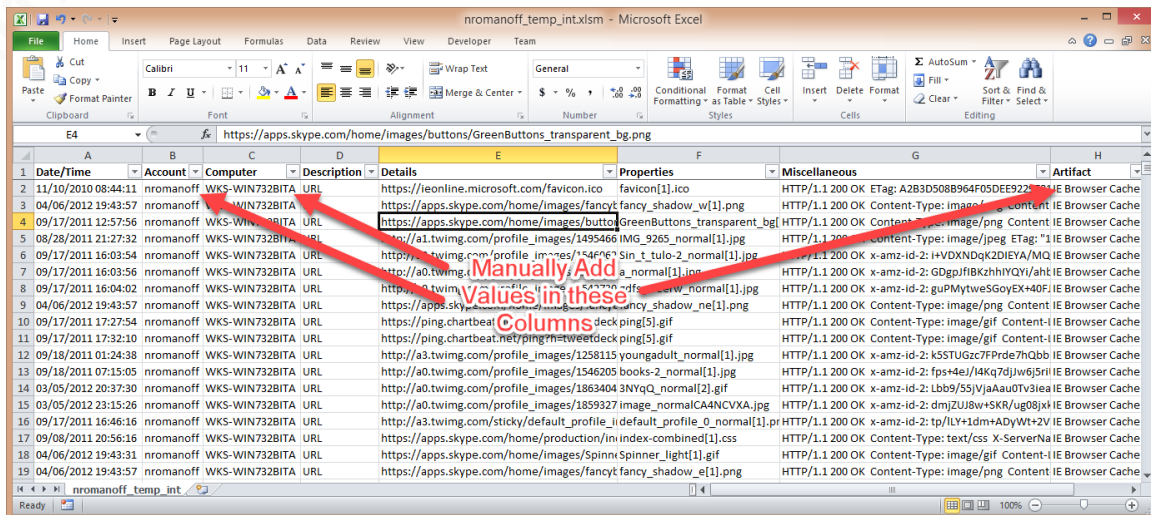
Figure 83. Final look. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

8.4.6. MFT Entries

We produced the MFT output used in this example by running the analyzeMFT python tool against a Windows 7 $MFT file:

C:\Python27\python.exe analyzeMFT.py -f $MFT -w –o MFT.csv

After running this command and importing the .CSV output file into Excel, you'll notice that there is a lot of data. You'll also notice that the Date/Time values are not in the correct format which will need to be corrected (Figure 84).

Figure 84. Correct Date/Time values. (Lee, 2014, digital case files)

You will delete most of the columns in this spreadsheet as they will not be used in the analysis. The only columns you should keep are those titled: Record Number; Filename #1; Std Info Creation date; Std Info Modification date; Std Info Entry date; FN Info Creation date; FN Info Modification date; and FN Info Entry date (Figure 85).
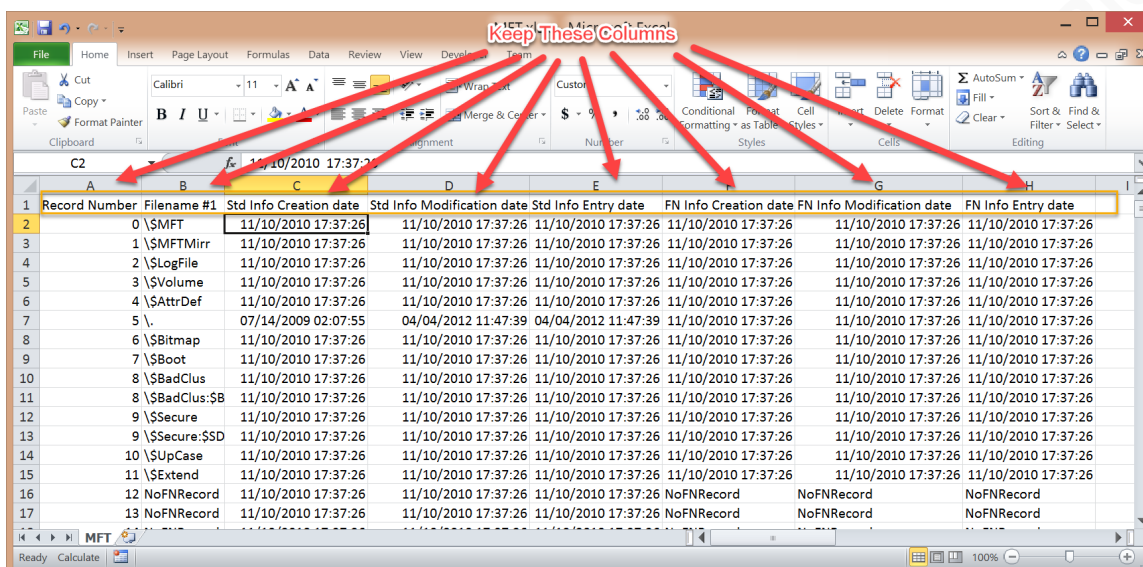
Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 85. Columns to keep. (Lee, 2014, digital case files)

The output from parsing the MFT file will most likely contain missing or corrupt information. You can identify rows containing this bad data by examining the 'Filename #1' column for cells that contain either the string "NoFNRecord" or "Corrupt MFT Record" (Figure 86).
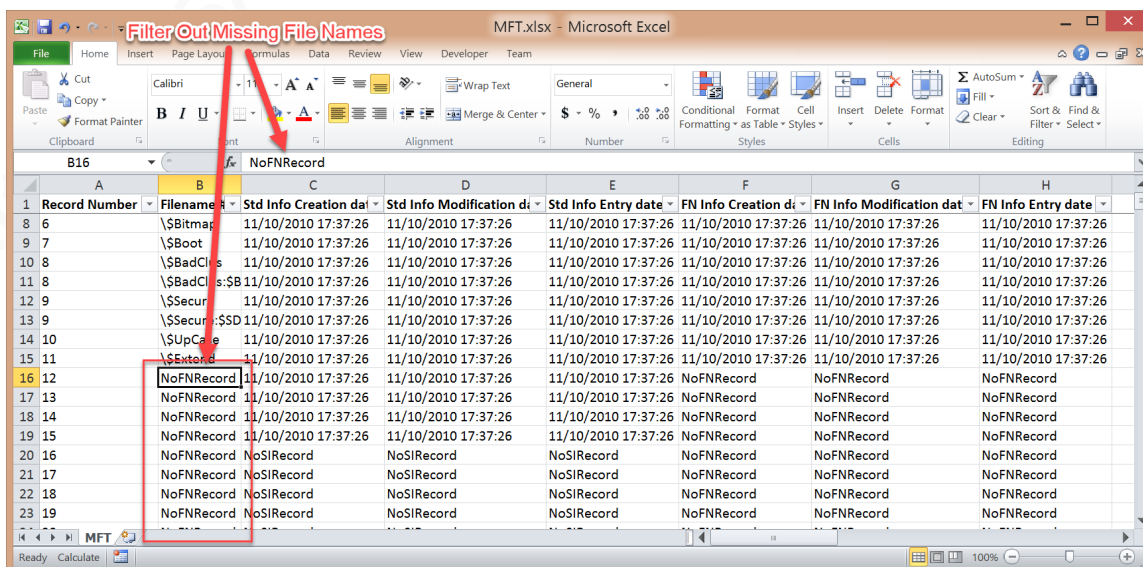

Figure 86. Corrupt data. (Lee, 2014, digital case files)

These entries can easily be removed by using the filter button on the 'Filename #1' column and filtering on those two phrases (Figure 87).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
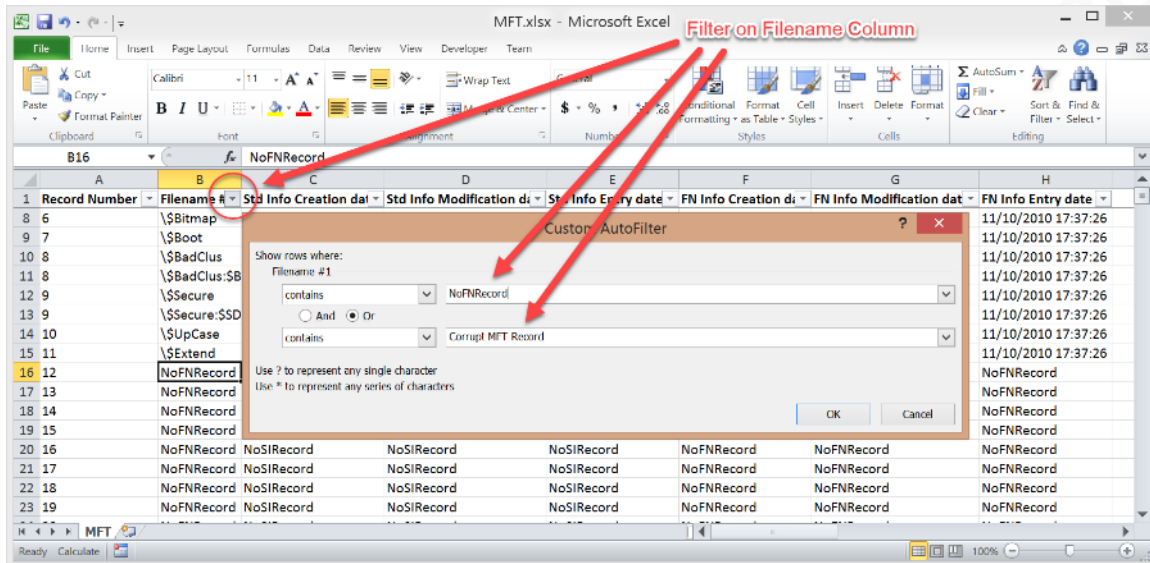
Figure 87. Filter out corrupt data. (Lee, 2014, digital case files)

Once you identify all the rows with those phrases, merely select them all and delete them (Figure 88).
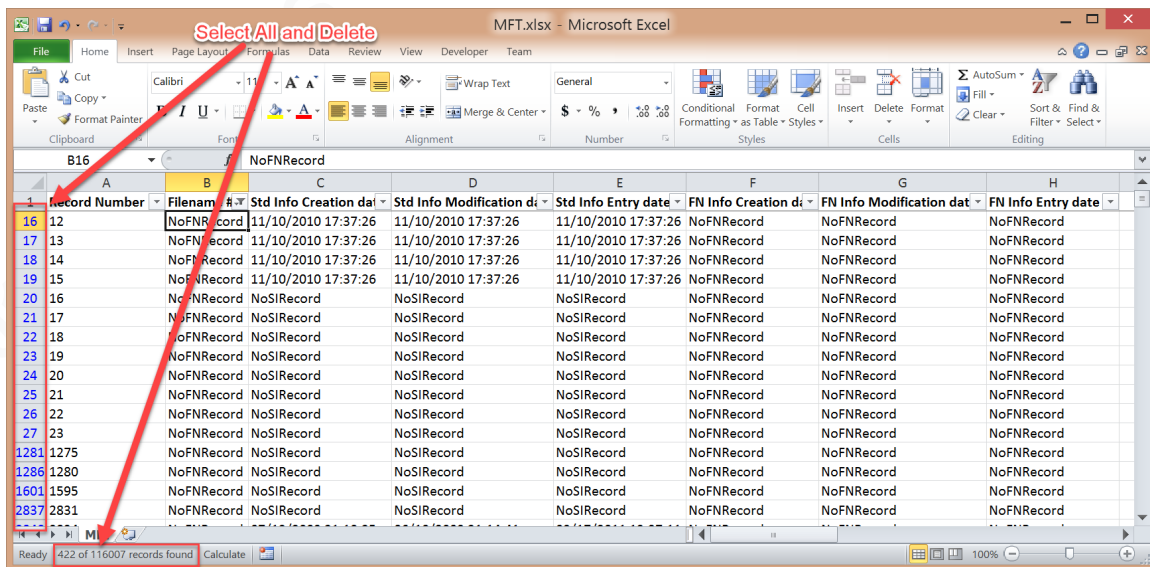

Figure 88. Delete corrupt data. (Lee, 2014, digital case files)

There are still too many columns to fit into our standard layout and too many timestamps to incorporate them all in a timeline chronology. To fix the layout, we select the 'FN Info Creation date' column to represent the chronology of the MFT entries, and we combine the remaining timestamps into two columns, one for the 'Std Info' timestamps and one for the remaining 'FN Info' timestamps (Figure 89).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
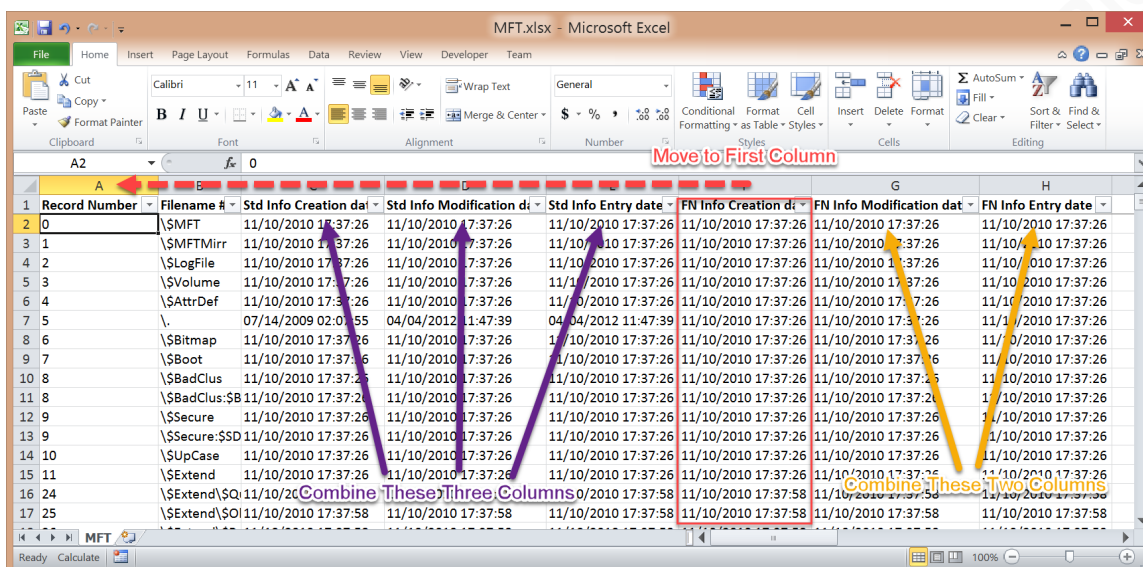
Hunting through Big Data with Excel | 6

Figure 89. Combining Date/Time columns. (Lee, 2014, digital case files)

To combine the columns of timestamps and to display those timestamps in the correct format, a function will be used that specifically tells Excel how the values should look. The first function will combine the 'Std Info' timestamps (Figure 90).

="Std Info - Create: " & TEXT(D2, "mm/dd/yyyy hh:mm:ss") & ", Modify: " & TEXT(E2, "mm/dd/yyyy hh:mm:ss") & ", Entry: " & TEXT(F2, "mm/dd/yyyy hh:mm:ss")
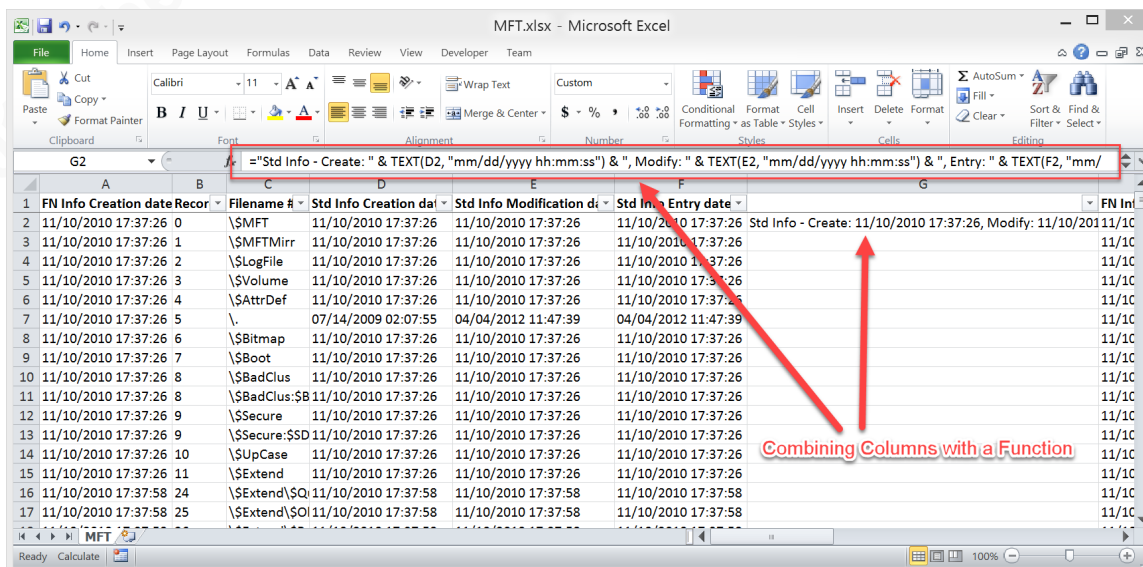

Figure 90. A function to combine columns. (Lee, 2014, digital case files)

The second function performs the same action, just on the 'FN Info' timestamps:

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

© 2017 The SANS Institute   Author retains full rights.

="FN Info - Modify: " & TEXT(G2, "mm/dd/yyyy hh:mm:ss") & ", Entry: " & TEXT(H2, "mm/dd/yyyy hh:mm:ss")

When dealing with a large set of data and filling a column with a function, you may run into an issue where the values in the cells do not match the formula applied to that cell (Figure 91 – Number 1). To fix this issue, on the 'Formula' ribbon, select the 'Calculate Now' button to refresh/recalculate the function formulas (Figure 91 – Number 2).
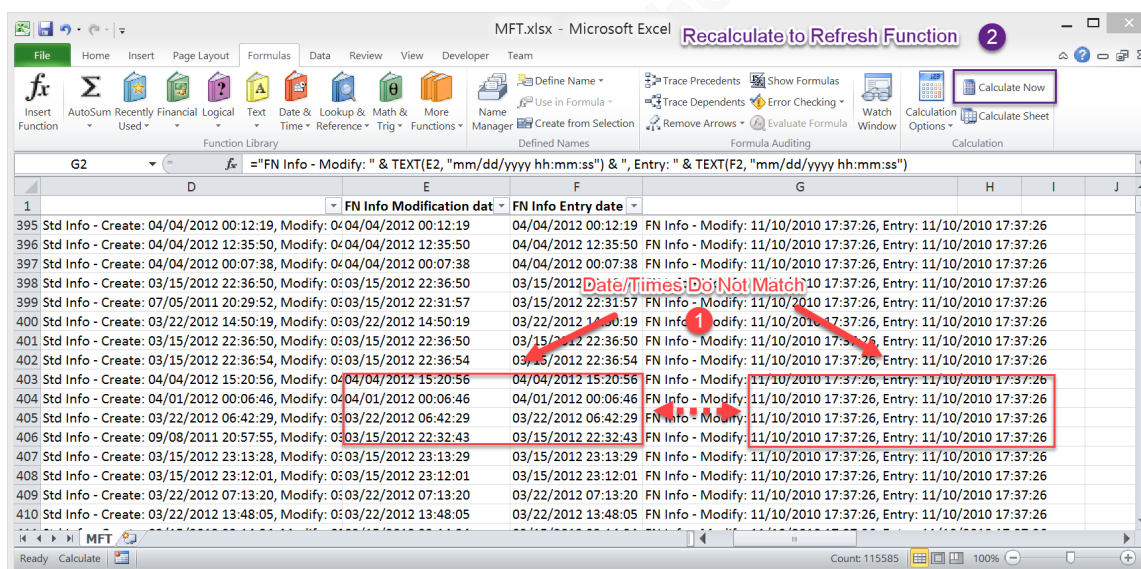

Figure 91. Recalculate function values. (Lee, 2014, digital case files)

The recalculated combined columns should look like Figure 92.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
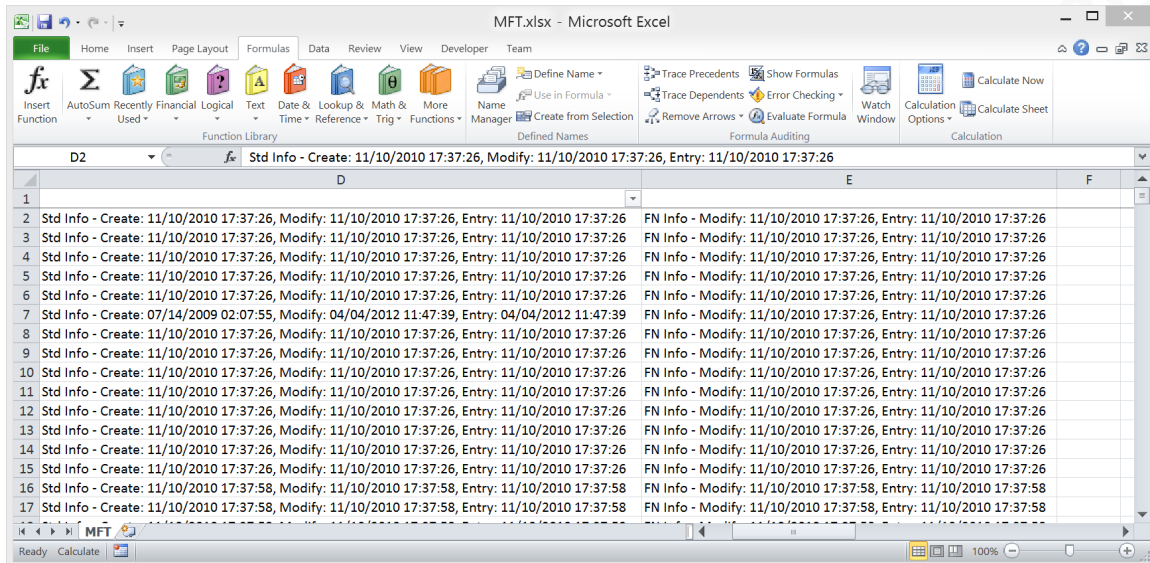
Figure 92. Date/Time column results. (Lee, 2014, digital case files)

A few final changes need to be made such as entering in the standard headers; creating and filling in the Artifacts column; moving the old 'Record Number' column to the 'Miscellaneous' column; and adding a prefix to the 'Miscellaneous' column for clarity (Figure 93).
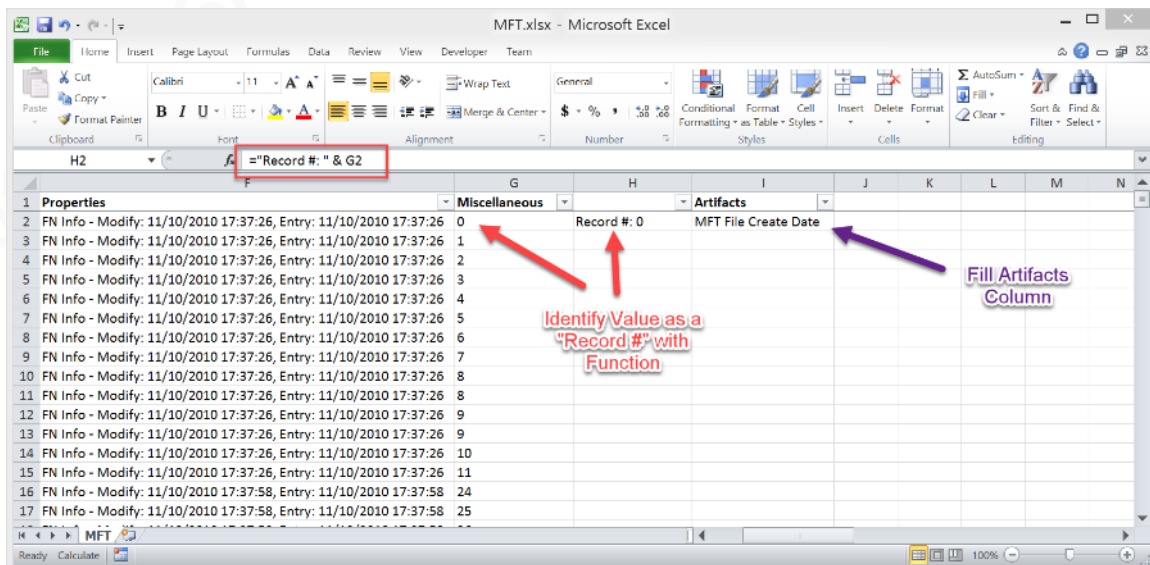

Figure 93. Moving 'Record Number' column and adding a prefix to columns. (Lee, 2014, digital case files)

The final output should look like Figure 94.

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 94. Final output. (Lee, 2014, digital case files)

8.4.7. Prefetch Entries

We produced the Prefetch output used in this example by running the parse_prefetch_info tool against the prefetch directory of a Windows 7 system:

parse_prefetch_info.exe –p <Path to Prefetch Directory> -d <database_name> -w Vista -o <Path to Output Directory> -r csv

One of the files generated when running the above command is prefetch_file_info.csv which contains a nice summary of the prefetch file activities on the system. When you open the file and apply the standard formatting, you'll notice that the 'UTC time' column is not formatted properly (Figure 95).

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User

Figure 95. Fix date and times to match our standard format. (Lee, 2014, digital case files)

Unfortunately, Excel does not recognize these values as Dates or Times, so we'll need to fix it so that it does. All the information is there, but the order in which it appears is not quite right. To correct this, we will reorder the data. First, we highlight the 'UTC time' column and then click the 'Text to Columns' button on the 'Data' ribbon to separate the items using a 'Space' character as the delimiter (Figure 96).
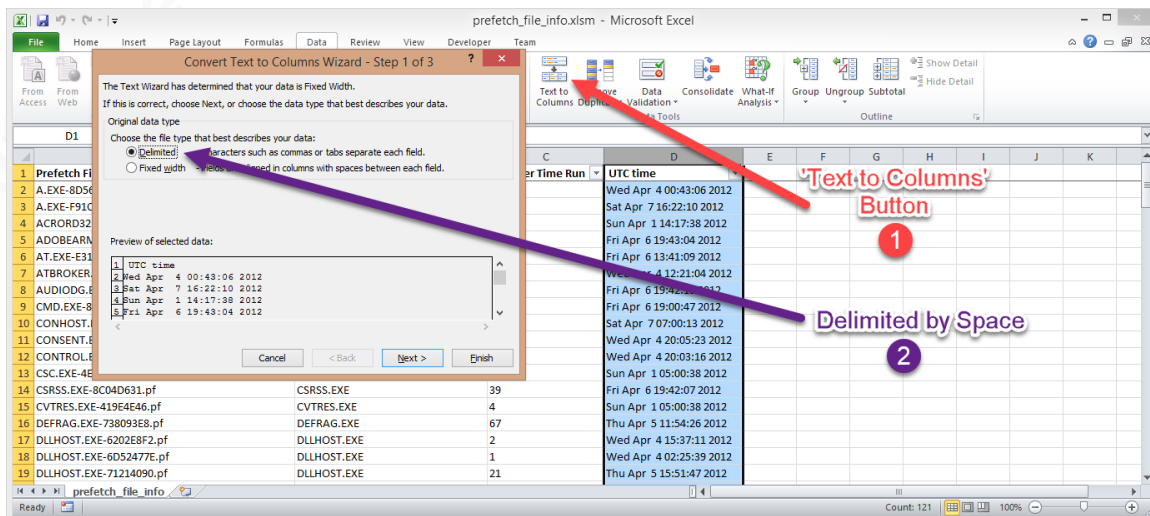

Figure 96. Text to columns to fix Date/Time. (Lee, 2014, digital case files)

Once each Date/Time item is in its own column, the following formula can be used to add them back together in the proper order (Figure 97):

=DATE(H2,MONTH(1&E2),F2) + G2

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User
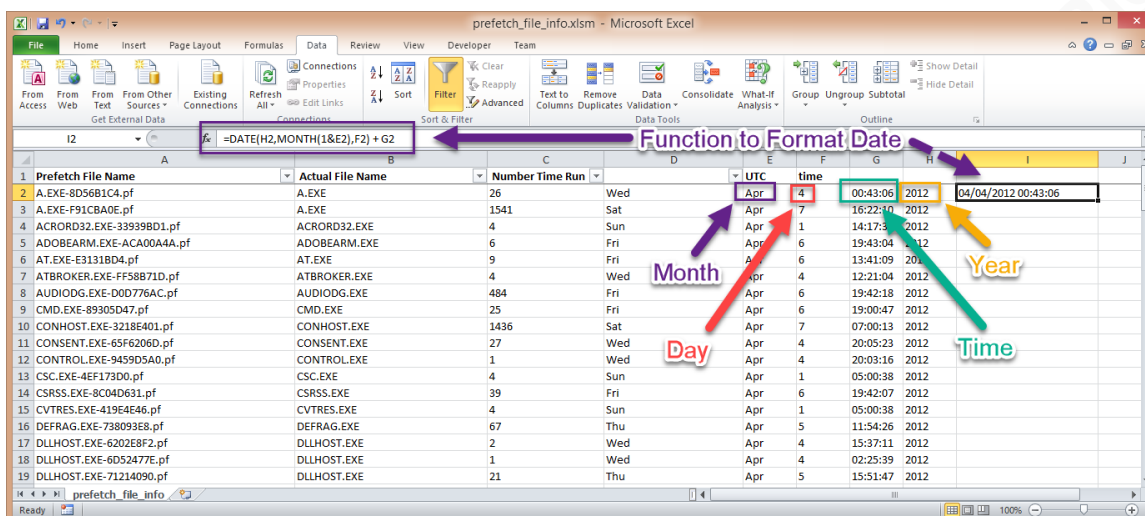
Figure 97. Reorder date and time fields. (Lee, 2014, digital case files)

After copying the cell values over the functions in the new column, you can delete the delimited columns you just created. Next, move the new Date/Time column to the beginning of the spreadsheet, insert columns where needed and modify the column headers to match our standard headers. Finally, prepend 'Number Time Runs' to the new 'Properties' column entries and manually enter the Artifact and Computer column values for a final result (Figure 98).
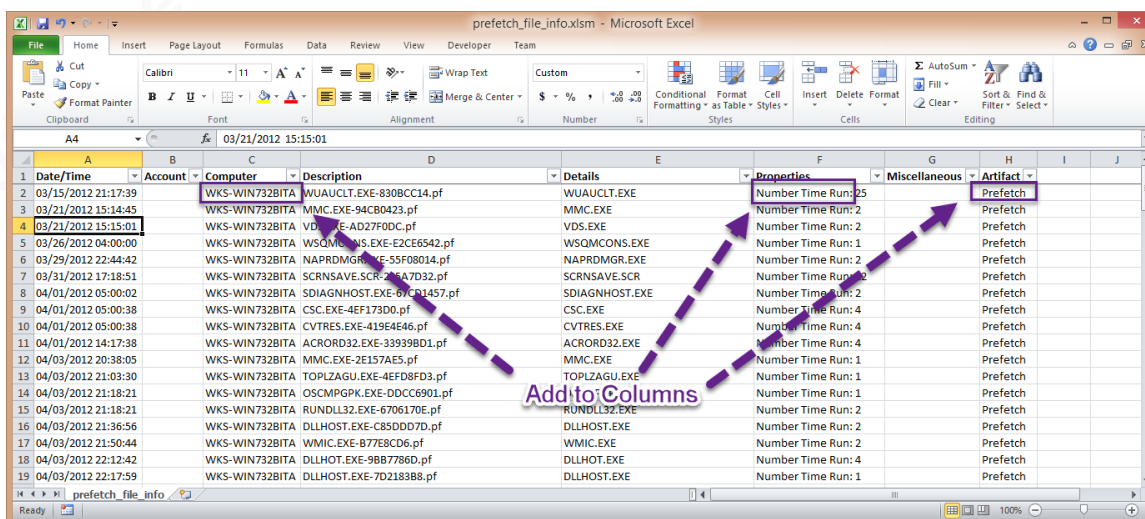

Figure 98. Prefetch results. (Lee, 2014, digital case files)

Microsoft Office UserMicrosoft Office UserGreg Lalla, greg.lalla@mail.comMicrosoft Office User