



A Testing Methodology for Rootkit Removal Effectiveness

Josh Harriman
Symantec Security Response
Dublin, Ireland

*Originally published by Virus Bulletin Conference, September 2007. Copyright held by Virus Bulletin, Ltd., but is made available courtesy of Virus Bulletin.
For more information on Virus Bulletin, please visit:
<http://virusbtn.com/>*

A Testing Methodology for Rootkit Removal Effectiveness

Contents

| | |
|------------------------------------|----|
| Abstract | 4 |
| Introduction | 4 |
| Testing Methodology | 4 |
| Tools | 5 |
| Test environment | 5 |
| Step-by-step testing | 5 |
| Details | 8 |
| File listings | 8 |
| Registry Modifications | 11 |
| Anti-Rootkit Tools | 12 |
| Special note - VISTA testing | 13 |
| Reporting the findings | 13 |
| Conclusion | 14 |
| References | 15 |

Abstract

Testing the effectiveness of an anti-rootkit product can prove difficult because of one simple fact. The threats you will be using to test these products will probably be hidden from most system monitoring tools. These tools are needed when evaluating anti-virus, anti-spyware or stand-alone anti-rootkit products, but could have little use against certain threats.

We need to consider taking a different approach when confronted with threats that hide their presence and modifications to the system under test. Using an offline discovery technique, we can find system changes that are made by these threats so we can successfully record their actions. This information is crucial when you are conducting an evaluation of one or more types of anti-rootkit products.

We will walk through this methodology and explain how to use the tools and gather the proper results. This technique should be used by independent testers when performing security product reviews against current rootkit threats.

Introduction

In order to test an anti-virus, anti-spyware or stand alone anti-rootkit product, you must be able to see what modifications were made to the system by the threat or risk. Normally this would be done using standard system monitoring tools such as Filemon, Regmon, and a before/after system snapshot tool (e.g. WhatChangedForWindows). However, a different methodology and sets of tools need to be used as we concentrate on testing threats that employ rootkit or stealth techniques. These techniques could have the ability to hide their modifications from standard monitoring tools. We need to make sure that we can capture any and all modifications made by these threats so we can conduct a thorough evaluation of a product's anti-rootkit capabilities in terms of detection and removal.

Using an offline approach on a suspect system is not a new concept, but we are going to cover it for use in a testing methodology. Booting a system with a known 'clean' BootCD is a way to load up the suspect file system so you can either scan it for infections, or in our case, look for side effects (files and registry modifications) associated with a particular threat. We will look at the tools available to accomplish this task.

It should be noted that this paper and techniques described herein are not for general discovery of rootkits or generic detection of new and unknown rootkits, but specifically for reviewing security products ability to detect and remove rootkits.

Testing Methodology

In this section we will discuss the testing methodology step-by-step. We will also discuss the tools needed to conduct an evaluation plus a proper testing environment. It's important to note that this testing methodology has some drawbacks depending on the type of threat under test. You will not be

able to test threats which are non-persistent. These are types of threats that do not leave persistent traces on the system after the system is rebooted. By traces, I mean registry keys, values and files which are intentionally left on the system so the threat will execute after reboot. New proof-of-concept threats such as Blue Pill¹, SubVirt², and PCI Rootkit³ also cannot be tested using this method.

Tools

- [Knoppix Live Linux BootCD](#)
 - [Helix Incident Response BootCD](#)
- [BartPE Live Windows BootCD](#)
 - BartPE will be used in the examples presented in this paper and during the step-by-step testing.
- [Alien Registry Viewer](#)
- Regedit/Regedit32 - Windows system executables
- [Streams](#)
- [File compare/diff program](#)
 - Windiff is a program that will be used in the examples presented in this paper.

Test environment

To ensure a repeatable testing environment, the test system should be imaged. Some imaging products available are [Norton Ghost](#) or [VMware](#). Be aware that some risks have the ability to determine if they are running in a VMware session. This could result in the risk not running at all, or give false results as to its behaviour. Using an imaged system is important for repeatable testing across multiple anti-rootkit products. The Operating System choice and setup of the test system should be similar to that of a typical user. A good base setup would be Windows XP Home Edition with Service Pack 2. This would suffice for most consumer users. If the testing is also to be conducted for the corporate edition of an anti-rootkit product, then other types of Operating Systems might need to be considered.

Step-by-step testing

1. Boot test system with one of the BootCD products listed in the tools section.
2. Copy the directory listing of the filesystem(s). This can be done with the following DOS command:

```
dir /s /a /o /b c:\ > [MOUNTED EXTERNAL DRIVE]\dos_results.txt.
```

The mounted external drive is there to help transfer files across to a separate system for analysis.

- You should also run a tool which can enumerate Alternate Data Streams (ADS). Some root kit threats employ these techniques and they will not be seen with the DIR command.

A Testing Methodology for Rootkit Removal Effectiveness

E.g. `Streams -s c:\ > ads_results.txt`

3. Copy the on-disk registry files, located here:

```
%Windir%\system32\config
```

You should also copy the user section file located here:

```
Documents and Settings\username\NTUSER.dat
```

4. Boot back to the test system and install/execute the rootkit sample(s).

5. Boot test system with one of the BootCD products listed in the tools section.

6. Copy the directory listing of the file system(s). This can be done with the following DOS command:

```
dir /s /a /o /b c:\ > [MOUNTED EXTERNAL DRIVE]\dos_results.txt
```

The mounted external drive is there to help transfer files across to a separate system for analysis.

- You should also run a tool which can enumerate Alternate Data Streams (ADS). Some root kit threats employ these techniques and they will not be seen with the DIR command.

E.g. `Streams -s c:\ > ads_results.txt`

7. Copy the on-disk registry files, located here:

```
%Windir%\system32\config
```

You should also copy the user section file located here:

```
Documents and Settings\username\NTUSER.dat
```

8. Boot back to the test system and run a full scan with the AV/AS product and record all results. Make sure to note any issues (app crash, blue screen, etc.) with the test system should they occur.

9. Boot test system with one of the BootCD products listed in the tools section.

10. Copy the directory listing of the filesystem(s). This can be done with the following DOS command:

```
dir /s /a /o /b c:\ > [MOUNTED EXTERNAL DRIVE]\dos_results.txt
```

The mounted external drive is there to help transfer files across to a separate system for analysis.

11. You should also run a tool which can enumerate Alternate Data Streams (ADS). Some root kit threats employ these techniques and they will not be seen with the DIR command.

E.g. `Streams -s c:\ > ads_results.txt`

12. Copy the on-disk registry files, located here:

```
%Windir%\system32\config
```

You should also copy the user section file located here:

```
Documents and Settings\username\NTUSER.dat
```

13. Determine changes made by the rootkit sample(s).
- Run file diff program on directory listing taken from step 2 and step 6. Record any files and/or directories added.
 - Run ARV (Alien Registry Viewer) and load the registry HIVES from step 3. Export this as a .reg file.
 - Run ARV and load the registry HIVES from step 7. Export this as a .reg file.
 - Run file diff program between the .reg export files taken in steps b and c above. Record any registry key/value modifications.
14. Determine if AV/AS product successfully removed the rootkit and its changes.
- Run file diff program on directory listing taken from step 6 and step 10. Record any files and/or directories removed.
 - Run ARV (Alien Registry Viewer) and load the registry HIVES from step 7. Export this as a .reg file.
 - Run ARV and load the registry HIVES from step 11. Export this as a .reg file.
 - Run file diff program between the .reg export files taken in steps b and c above. Record any registry key/value modifications.

Details

In this section, we will discuss in detail some of the methods listed in the step-by-step testing. We also need to understand the limitations to using ARV when viewing the registry modifications and discuss some changes to the testing method when you conduct an evaluation on Microsoft Vista Operating System.

File listings

Some threats that employ rootkit or stealth techniques will normally hide their files from the Explorer window (process) and the internal DOS command DIR. The threat could use various techniques to accomplish this⁴. But as long as the files remain on the system between reboots, we should be able to find them with the offline analysis approach. There are a couple of exceptions and we will discuss these as well.

We can boot the system with BartPE and run the DIR command with the following flags:

- /S for iterating through all directories and subdirectories
- /A for finding all files regardless of their attributes settings.
 - This will help find files which use a basic hiding technique of flipping on the hidden attribute flag.
- /B for displaying the results as Bare. This will help when using the file diff program.
 - There is a chance of missing file infector threats by using this flag.
- /O for displaying the results in order (alphabetically). Again, this will help when using the file diff program.

The screen shot shows how it could be difficult to find changes with Windiff when not using the /B flag. You want to be able to find the changes - red = removed, yellow = added - easily. This shot also shows that nothing has changed here but the timestamp.

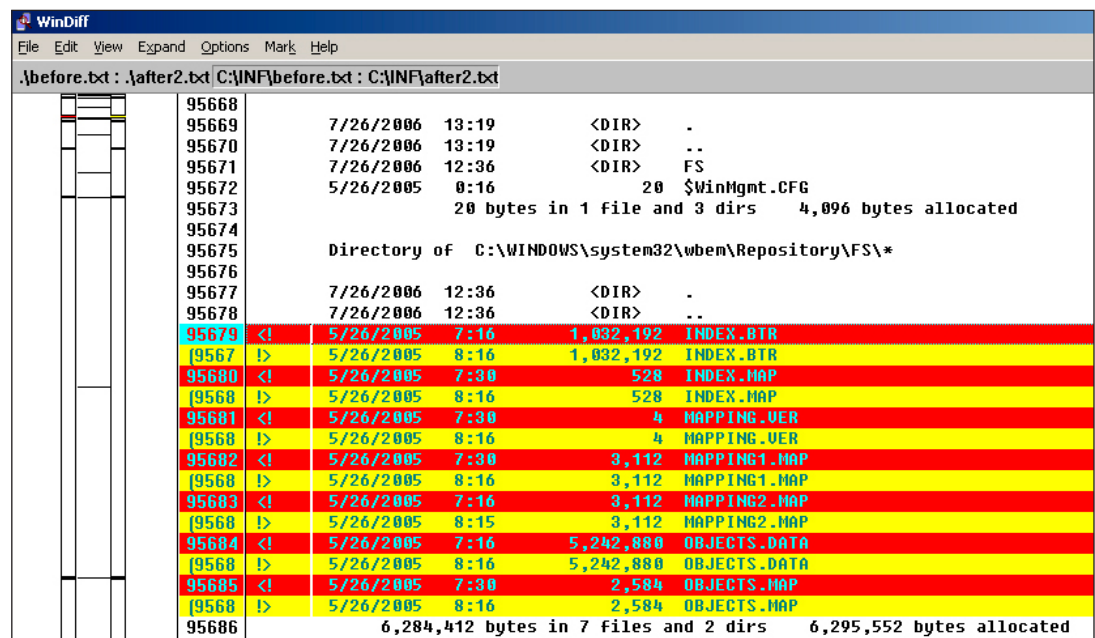


Figure 1: WINDIFF without /B flag

We also should run and ADS aware tool as mentioned in the step-by-step process.

Streams -s. The -s flag is for a recursive such.

One such rootkit threat which employs the ADS technique is Backdoor.Rustock A and B variants⁵. In this case, you can see below that the ImagePath for the service that Rustock creates is pointing to %System%:[RANDOM NUMBER]. The ':' denotes that this path is for a stream inside the %System% folder. The ADS technique is not really an exception and an ADS aware tool should be run each time.

One case that should be mentioned as an exception is file infector threats. If a threat was to inject its code into a file already on the system, the technique above using the /B flag would miss this change. Although the occurrence of this is low, it should nonetheless be noted. Running the DIR command without the /B flag could make the Windiff results rather difficult to sort and distinguish the changes.

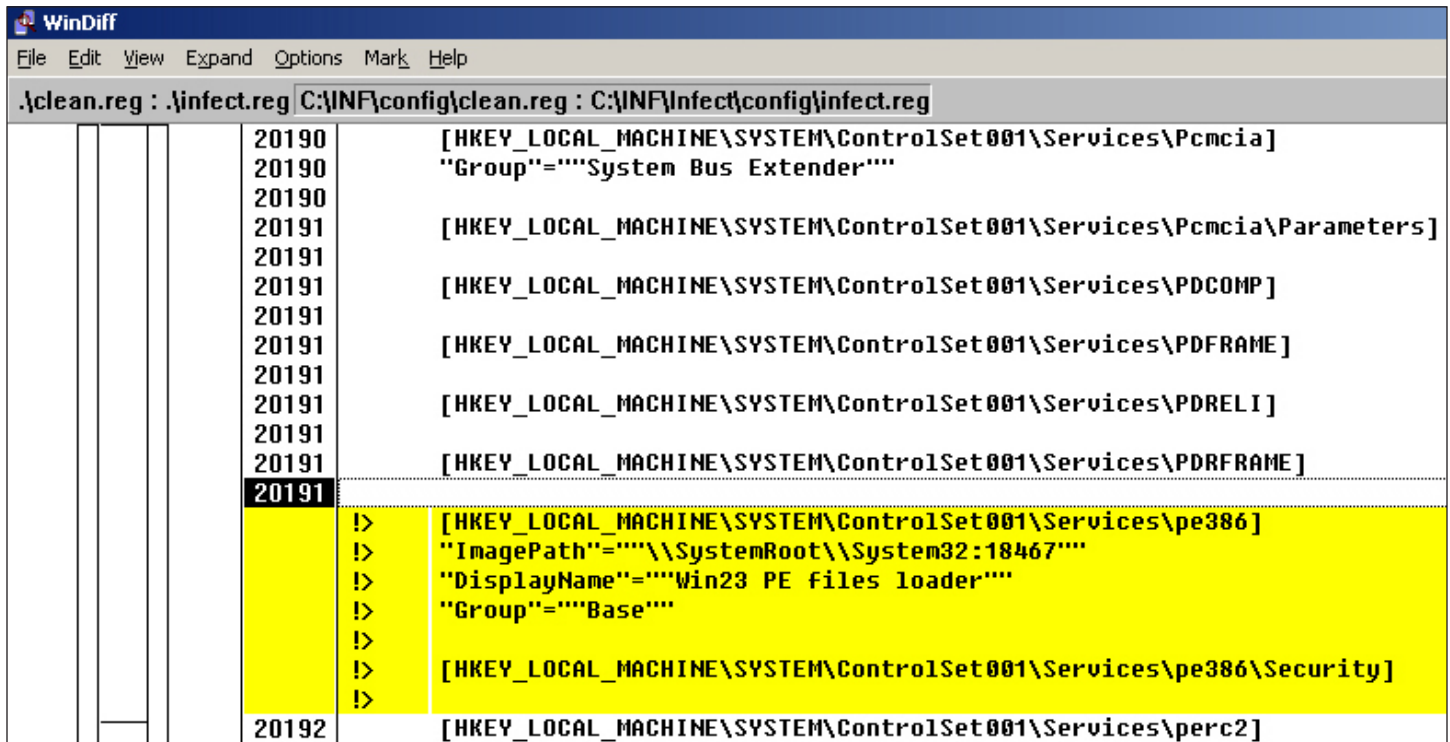


Figure 2: Rustock registry entry

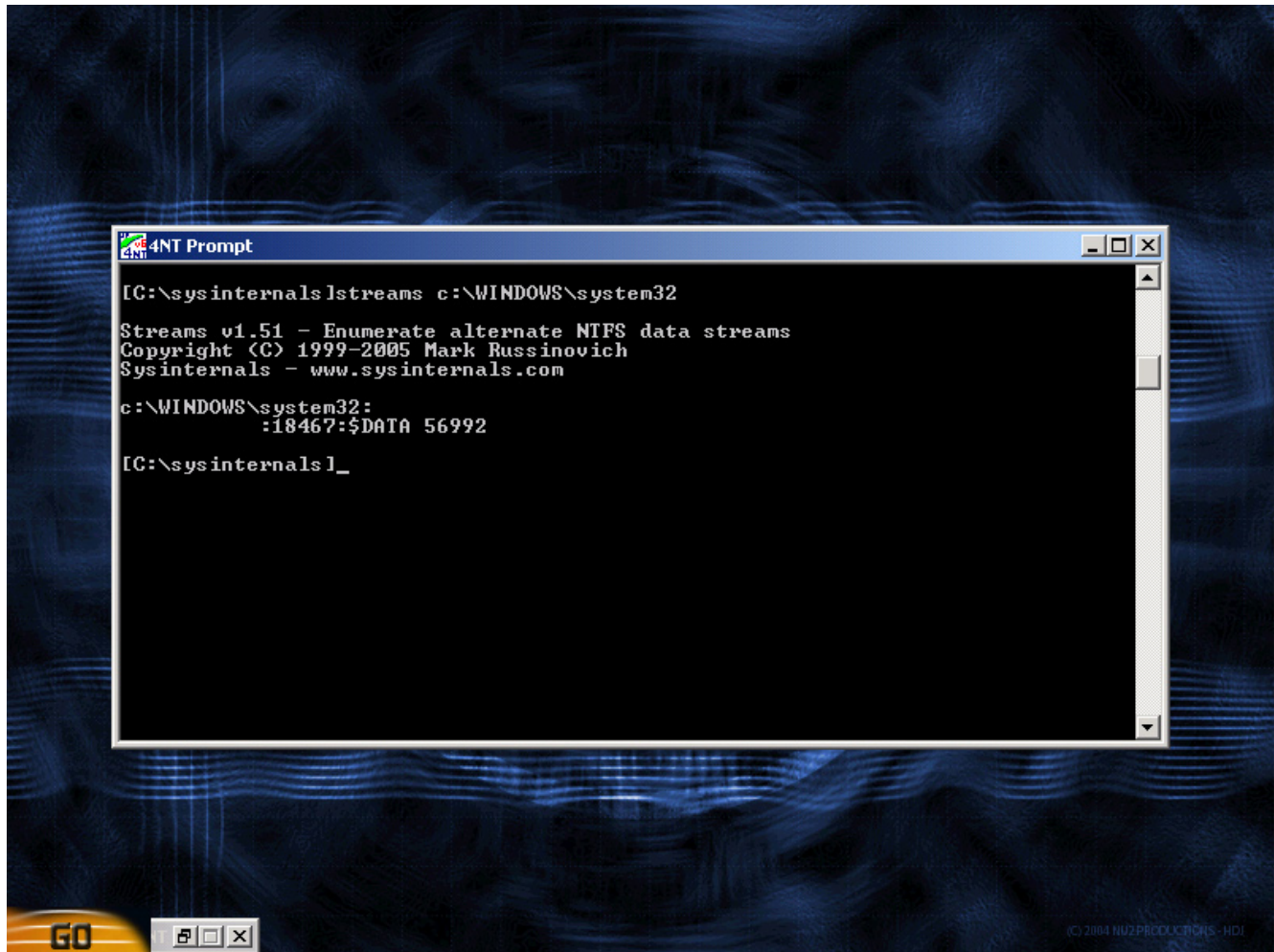


Figure 3: Streams tool running in BartPE against Rustock.

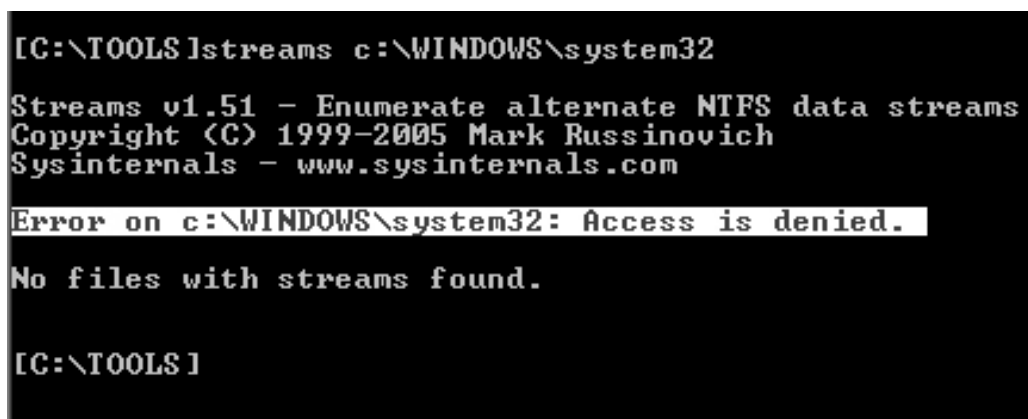


Figure 4: Streams tool running on live system against Rustock.

Registry Modifications

If a threat is to be successful in surviving a system reboot, they will most likely need to add some information to the registry. There are numerous locations that could help the threat restart after a reboot, and we will list some of the most common below. Most of these are known as Windows load points.

- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
The most common load point in Windows is the Run key in the registry. Files located in this key will execute after system startup.
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects
“In its simplest form, a BHO is a COM in-process server registered under a certain registry key. Upon startup, Internet Explorer looks up that key and loads all the objects whose CLSID is stored there”⁶
- HKEY_LOCAL_MACHINE\Software\Classes\CLSID\[CLSID NUMBER]\InprocServer32
A path to the file, which is loaded as a COM object, will be located in this key.
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
 - The Shell value by default will have Explorer.exe as the process that loads during system startup. Some risks will add a path to their file in this location.
 - The Userinit value by default will have userinit.exe as the process that loads during system startup. Some risks will add a path to their file in this location.
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
Another location in the registry that will load files after system startup.
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Runonce
- HKEY_CLASSES_ROOT\exefile\shell\open\command
The (Default) value could be changed to load a suspect file every time an .exe file is executed on the system.
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services
A location that can be added to start a threat as a Service on the system.

Since we are using an offline approach for testing, we need to be able to gather and read the registry HIVES⁷ from the test system. There are two programs that can read offline registry HIVES. Both are mentioned in the Tools section and we will go into detail about both below.

ARV, or Alien Registry Viewer can load the files that make up the registry HIVES. Once loaded, we can then export this 'snapshot' of the registry into a .reg file. The benefits to this are we will have one .reg file for each testing point.

1. Clean/Base line snapshot
2. Infected snapshot
3. Removed snapshot

This .reg file will be approximately 2MB in size. This will make it easy to use a file diff program to gather our results. The downside to using ARV is that it will only export REG_SZ data type (used for Registry values containing strings) values along with keys and subkeys. It's possible that a threat might make modifications to the registry but not as REG_SZ data type. We can see the other data types and changes within the ARV GUI, but that does not help us find the changes to begin with.

In order to see these other data types (plus the REG_SZ of course) within a .reg export file, we need to use either REGEDIT or REGEDIT32. But of course there is a slight downside to using this technique as well. You can only load and export one HIVE at a time. Then you would need to run your file diff program on each of these .reg files for each of the 3 testing points mentioned above. Also, the .reg file for one HIVE could be more than 20MB in size. The steps to load an offline gathered HIVE in REGEDIT is listed below.

1. Highlight `HKEY_LOCAL_MACHINE`

2. Choose File > Load Hive..c

This will open a dialog box for you to find and select one of the 5 HIVES to open.

3. You will need to name it something other than ones that are currently loaded. For example, if you are loading SOFTWARE, you could rename it to CleanSoftware (for the one you generated from testing point 1 above).

4. Once all of these are loaded you can highlight each one and export them.

5. Before you run your file diff program, open the file and rename CleanSoftware to SOFTWARE. This needs to be consistent across all of the files because the diff program will see every line as different, instead of the ones we are interested in.

Anti-Rootkit Tools

In the tools section we do not mention anti-rootkit analysis tools. There is a large list of [available tools](#)

but we do not mention using them during our testing methodology. The main reason behind not using these tools is that quite a few of them are being bypassed explicitly by the rootkit threats themselves. Some of these techniques are easier than others, but since it's not possible to know if the threat (without further static analysis on the file/threat itself) is trying to do this, I believe it is better and safer to bypass using them during the testing phase.

Special note - VISTA testing

Currently the ARV program cannot read the registry HIVES from VISTA. That part of the test will have to be done with REGEDIT or REGEDIT32 as discussed at the end of the Registry modifications section.

Reporting the findings

We have talked about the offline analysis technique and how to use the tools, but we need to also think about how to report the findings if you are conducting an evaluation of an anti-rootkit product. All of the information that comes from following the step-by-step testing and the detail section above are not needed in the final report. The report needs to be concise yet complete. A good report should include the following:

1. Common name of rootkit samples used for testing

Most threats and risks have common names that are shared among the anti-virus industry.

2. Files and registry entries added or modified by the installation of the sample

These files can be grouped together as totals into meaningful buckets for critical and non-critical. The same could be applied to the registry keys and values following the load point section mentioned above.

3. Product information

- The exact version of the anti-rootkit product.
- The exact version of the data file definitions. If versioning information is not available, then the exact date and time the definitions were updated.

4. Files and registry entries deleted or modified by the anti-rootki product

- These files can be grouped as totals into meaningful buckets for critical and non-critical. The same could be applied to the registry keys and values following the load point section mentioned above.
- This could also be represented as a percentage.

5. Any issues with the test system or applications after the full scan is completed

This would be detail from Step 7 in the step-by-step testing section.

6. An appendix that list details of the rootkit samples

This would include the location of the sample e.g. the Internet address, the MD5 value of the samples and the date and time each sample was harvested.

A rating scale, such as the following could be used to show the removal effectiveness of the anti-rootkit product:

- 1. Poor:** Not enough critical files, processes, load points removed. Rootkit risk still exist on the system.
- 2. Good:** All critical files, processes, load points removed. All rootkit risks were neutralized on the system. Some remnants are left behind. These remnants could be non-critical registry keys/values, log files, empty directories.
- 3. Best:** All critical files, processes, load points and remnant side effects are removed. This would be a complete removal of all files and registry entries created by the rookit sample.
- 4. Aggressive:** This rating could be given to an anti-rootkit product that removed too much from a system or incorrectly removed some risks. This would include incorrectly removing LSP registry entries, or registry entries of a legitimate application and their removal caused system instability, or application crashes.

Conclusion

Testing anti-rootkit enabled products need a slightly different testing method to measure their effectiveness. It's not so much the product, but how to interpret the system changes made by the threats themselves that is important. All the tools mentioned are freely available and relatively easy to use.

Remember to record your results concisely and thoroughly. It's important to keep to all the information gathered during the testing in case you are ever called upon to back up your findings. Not matter how solid your testing methodology is, the results and reporting phase are extremely important.

Using this testing methodology should help you achieve a proper evaluation of any anti-rootkit enabled product. Whether you are an independent product tester, or a company employee in charge of product evaluations, this testing methodology will help you to test rootkit type threats.

References

1. Rutkowska, J (2006). Subverting Vista Kernel For Fun and Profit. Retrieved February 20, 2007 from <http://invisiblethings.org/papers.html>.
2. Univ Michigan and Microsoft Research et al. (2006). SubVirt: Implementing malware with virtual machines. Retrieved February 20, 2007 from <http://www.eecs.umich.edu/virtual/papers/king06.pdf>
3. Heasman, J. (2006). Implementing and Detecting a PCI Rootkit. Retrieved February 20, 2007 from http://www.ngssoftware.com/research/papers/Implementing_And_Detecting_A_PCI_Rootkit.pdf
4. Hoggland, G. and Butler, J. (2005). Rootkits: Subverting the Windows Kernel, Addison-Wesley Professional. Retrieved February 20, 2007.
5. Symantec (2006). Backdoor.Rustock. Retrieved February 20, 2007 from http://www.symantec.com/security_response/writeup.jsp?docid=2006-060111-5747-99
6. (Esposito, 1999). Browser Helper Objects: The Browser the Way You Want It. Retrieved February 20, 2007 from <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebgen/html/bho.asp>.
7. Hipson, P. (2000). Mastering Windows XP Registry, SYBEX Inc. Retrieved February 20, 2007.

About Symantec

Symantec is the global leader in information security, providing a broad range of software, appliances, and services designed to help individuals, small and mid-sized businesses, and large enterprises secure and manage their IT infrastructure.

Symantec's Norton™ brand of products is the worldwide leader in consumer security and problem-solving solutions.

Headquartered in Cupertino, California, Symantec has operations in 35 countries.

More information is available at www.symantec.com.

Symantec has worldwide operations in 35 countries. For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 800 745 6054.

Symantec Corporation
World Headquarters
20330 Stevens Creek Boulevard
Cupertino, CA 95014 USA
408 517 8000
800 721 3934
www.symantec.com

Symantec and the Symantec logo are U.S. registered trademarks of Symantec Corporation. Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other brand and product names are trademarks of their respective holder(s). Any technical information that is made available by Symantec Corporation is the copyrighted work of Symantec Corporation and is owned by Symantec Corporation. NO WARRANTY. The technical information is being delivered to you as-is and Symantec Corporation makes no warranty as to its accuracy or use. Any use of the technical documentation or the information contained herein is at the risk of the user. Copyright © 2007 Symantec Corporation. All rights reserved. 04/05 10406630