5

REGRIPPER

INFORMATION IN THIS CHAPTER

- What is RegRipper?
- Getting the most out of RegRipper

Introduction

Since it was first released, RegRipper has been downloaded a great number of times and seems to be used by a great many analysts. However, I tend to wonder just how many analysts really *use* RegRipper to get the most from the Registry hives they're examining, as opposed to those that simply run the tool because they heard someone say that they should. There's much more available to an analyst when employing a tool such as RegRipper, and the purpose of this chapter is to provide a foundation not only of how to employ RegRipper, but also how to get the most out of using RegRipper.

Over the years, I've written a number of blog posts and discussed RegRipper in a number of forums (conference presentations, etc.). In some cases, I've written blog posts to answer individual questions I've received, and in other cases, I've written blog posts to address somewhat more "collective" questions, the kinds of things I see over and over again. My intent for adding this chapter to this book is to consolidate a lot of that information in one location.

What Is RegRipper?

From our discussion in chapter Processes and Tools, you could say that RegRipper is a "superparser" for the Windows Registry. RegRipper is a framework that runs various plugins, individually or in groups, where the plugins are used to extract specific data from within the Registry hive files, parse and translate that data as necessary, and then display the results for the analyst. RegRipper is not a viewer application; you cannot load a hive file into RegRipper and browse through the structure. What you can do is parse binary value data to display contents that would not

Windows Registry Forensics. http://dx.doi.org/10.1016/B978-0-12-803291-6.00005-X

be visible in a viewer application. For example, you can decode such things as ROT-13 encoded value names for display, and you can format parsed and decoded data in whatever manner fits the needs of your analysis.

Tip

In the fall of 2014, I decided to put RegRipper in one easy-to-reach location, and I created a project page on GitHub. If you're at all interested in the latest, most up-to-date version of RegRipper or any of the plugins, you can get them from https://github.com/keydet89/RegRipper2.8.

RegRipper started its life as a couple of Perl scripts I'd written in order to retrieve specific data from hives during exams. I had thought at the time, why should I continue to open the hive file in a viewer and navigate to the key or value of interest when it's quicker to just have a computer program do it for me? So a couple of Perl scripts became a couple more, and then a few more, and pretty soon I was getting to the point where I was running out of short names for the scripts. I was also getting to the point where I felt like I needed some way to better manage the scripts, not only just keeping them but keeping track of which script extracted what data. The idea I came up with was to continue using the Perl programming language but develop a way to combine all of the common elements of the scripts into a framework so that I didn't have to continually rewrite those elements. The result was a command line tool that would run the various scripts, but in no particular order. Over time, I added a graphical user interface (GUI), and the ability to run groups of scripts, or "plugins," in a specific order. Since RegRipper was first released, a number of plugins have been added, and others have been removed or modified.

RegRipper itself is written in Perl, and the plugins are Perl code as well. RegRipper is distributed as Perl source code, and the archive also includes Windows executable files for both the RegRipper GUI tool, as well as the command line tool (rip.exe), both of which were "compiled" using Perl2Exe (found online at http://www.indigostar.com/perl2exe.php). As such, the dynamically linked library (DLL) file that is included with the distribution, named p2x5124.dll, must be in the same folder as the .exe files for RegRipper (rr.exe, rip.exe) so that the applications can be successfully launched.

RegRipper is not only available as a standalone download, but it is also available in Linux-based forensic application distributions

such as PlainSight (found online at http://www.plainsight.info/features.html) and the SANS Investigative Forensic Toolkit (aka SIFT), which can be found online at http://digital-forensics.sans.org/community/downloads. Slight modifications to the original code for the command line component of RegRipper allowed it to also run on Linux systems; these modifications were incorporated into the RegRipper GUI as well. RegRipper is also included in the Autopsy framework (found online at https://github.com/sleuthkit/autopsy).

Most analysts utilize the RegRipper GUI, by either opening a command prompt, navigating the directory where RegRipper was copied, and typing "rr" at the command line, or by double-clicking the yellow "pearl" (not "Perl," "pearl") icon for "rr.exe". Either one will open the RegRipper GUI, which is illustrated in Fig. 5.1.

Once the RegRipper GUI appears, you select a hive file by clicking the "Browse" button to the right of the "Hive File" text field, and an output report file, and then select the appropriate profile

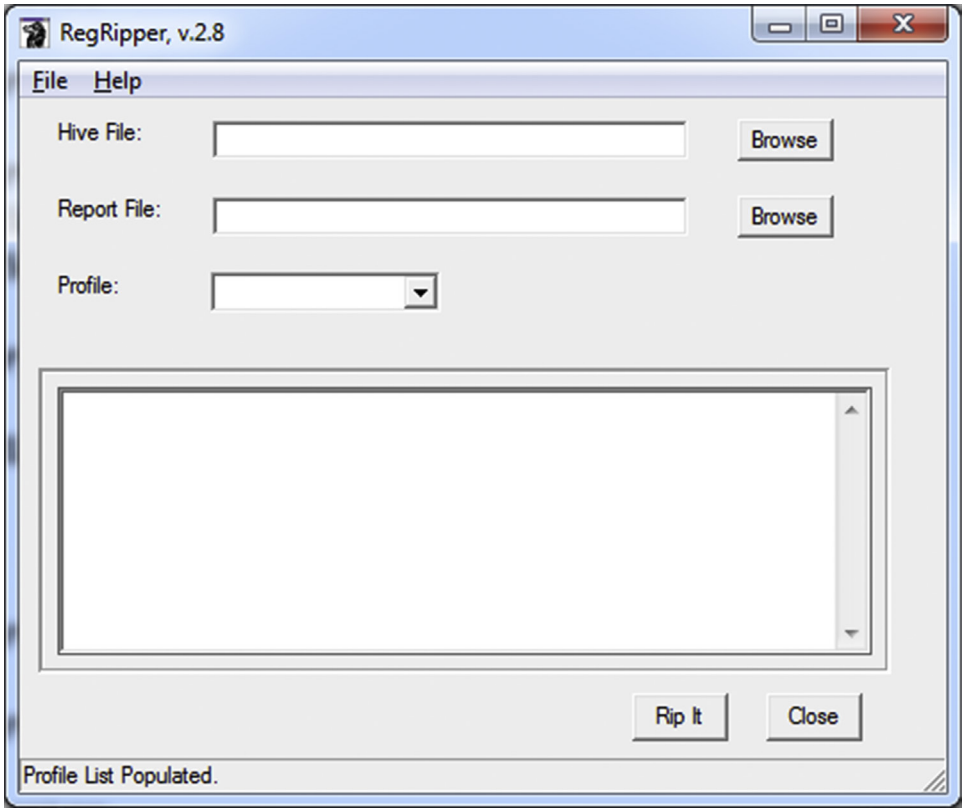

Figure 5.1 RegRipper GUI.

from the "Profile:" drop-down list. Clicking the "Rip It" button will run the plugins within the selected profile, in order, against the hive file, write the output of each plugin to the report file, and report progress information in the large text field.

When RegRipper completes running all of the plugins against the selected hive file, you will see something similar to what is illustrated in Fig. 5.2.

As you can see in Fig. 5.2, RegRipper displays a running status of the plugin being run in the large text window, and when the tool has completed running all of the plugins in the profile, the word "Done" appears in the status bar at the bottom of the main window. You can also see in Fig. 5.2 that the tool reported that "4 plugins completed with errors." What many folks don't realize is that when the RegRipper GUI is used, this tool maintains a log file of its own activity. In the "Report File" text field, you can see that the output of all of the plugins will be written to a file named "D:\cases\jason\test_ntuser.txt". Once the tool has completed, you will also find a file named

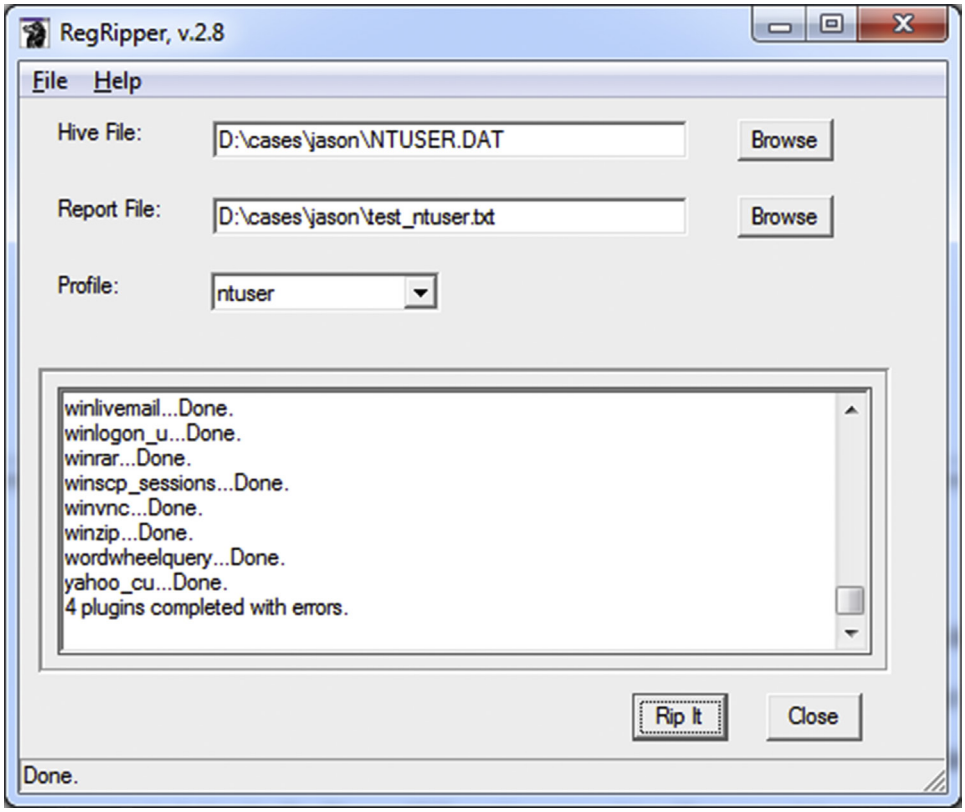

Figure 5.2 Description.

"test_ntuser.log" in the same folder; this is the file that contains the log of activity. This file is intended to log the hive file that RegRipper is being run against, each hive file that is run against the hive file, and any significant errors. This log file maintains a record of which plugins (including the version of each plugin) were run against the hive and allows the log file to be included as part of an analyst's case notes. Each line of the log file starts with the date/time stamp of the entry. If a plugin encounters an error, that error is also recorded in the log file and can be used for troubleshooting purposes.

Warning

The RegRipper GUI allows you to run groups of plugins, or profiles, against a hive file. It doesn't allow you to run individual plugins against a hive file, unless you want to use a profile that contains just one plugin. The capability for running one plugin against a hive file wasn't included in the RegRipper GUI, as it's not terribly efficient, and the capability exists with the command line RegRipper component (rip.exe).

Plugins

Plugins are small Perl scripts that are run by RegRipper and perform specific functions. These are the bits of code that do all of the work of RegRipper: accessing the appropriate key path(s), extracting metadata from the key structures, accessing values, parsing and displaying data, etc. As they are Perl scripts, plugins end with the ".pl" file extension.

Tip

By design, if you are running RegRipper by using the Windows executable files (rr.exe, rip.exe), you do not need to have Perl installed on your system. Even though the plugins are Perl code and are not "compiled" in the same manner as the main RegRipper components, you still do not need Perl installed in order to run the complete set of plugins.

Most plugins are intended for specific hives; the data that they collect are only found in one hive. In some cases, the data that you may be looking for may be found in either the Software hive or the NTUSER.DAT hive, with the only difference in extracting the desired data from one hive or the other is a slight variation in the path to that data. In cases such as these, it is trivial to have a plugin that can be run against either hive file and "fail silently" (that is, not report an error) if one path does not succeed in finding the data.

Some plugins are intended to extract very specific data, making them very "tactical," while others are more general or "strategic" in nature. For example, the regin.pl plugin checks the System hive file for specific values known to be indicative of the Regin malware. The renocide.pl plugin checks the Software hive for a specific key known to be indicative of the Renocide malware. These plugins are very "tactical," as they check for specific keys or value data. On the other hand, the svc.pl plugin accesses the Services key in the System hive and extracts (and displays) data from all of the subkeys, making no attempt to filter the data.

Some plugins can be run across all hives. For example, the findexes.pl plugin was written to look for binary data types (values with data of type 0 or 3) that start with "MZ", the first 2 bytes of a Windows portable executable file. The rlo.pl plugin was written to check any hive file for any key and value names that contain the right-to-left override Unicode control character. The fileless.pl plugin was written to check any hive file for indications of the Poweliks malware; throughout the growth of the malware, it employed different persistence mechanisms that spanned multiple hives, while the value data itself remained similar enough to be detected through the use of a simple regular expression.

Profiles

Profiles within RegRipper are found in the "plugins" folder and are not Perl code. Rather, profiles are text files with no extension; for example, rather than "profile.txt", they are named simply "profile". These files contain just a list of plugins to be run, one on each line. If you choose to run RegRipper using a profile, each of the plugins listed in the profile is run in the order in which they appear in the file.

When you launch the RegRipper GUI, you'll be able to see the available profiles by exposing the drop-down menu for the "Profile:" listbox, as illustrated in Fig. 5.3.

You can create your own profiles for your own use or to share with others. I know several folks who've done this, and I've done it upon occasion. I don't include those in the RegRipper distribution because I've found that it tends to confuse some of those who download and run RegRipper; with too many plugins available, there appears to be confuse as to which ones to run. However, we do discuss how to create your own profiles in detail later in this chapter. Creating your own profile is as easy as opening Notepad and typing the name of each plugin that you want to run against a hive file on one line.

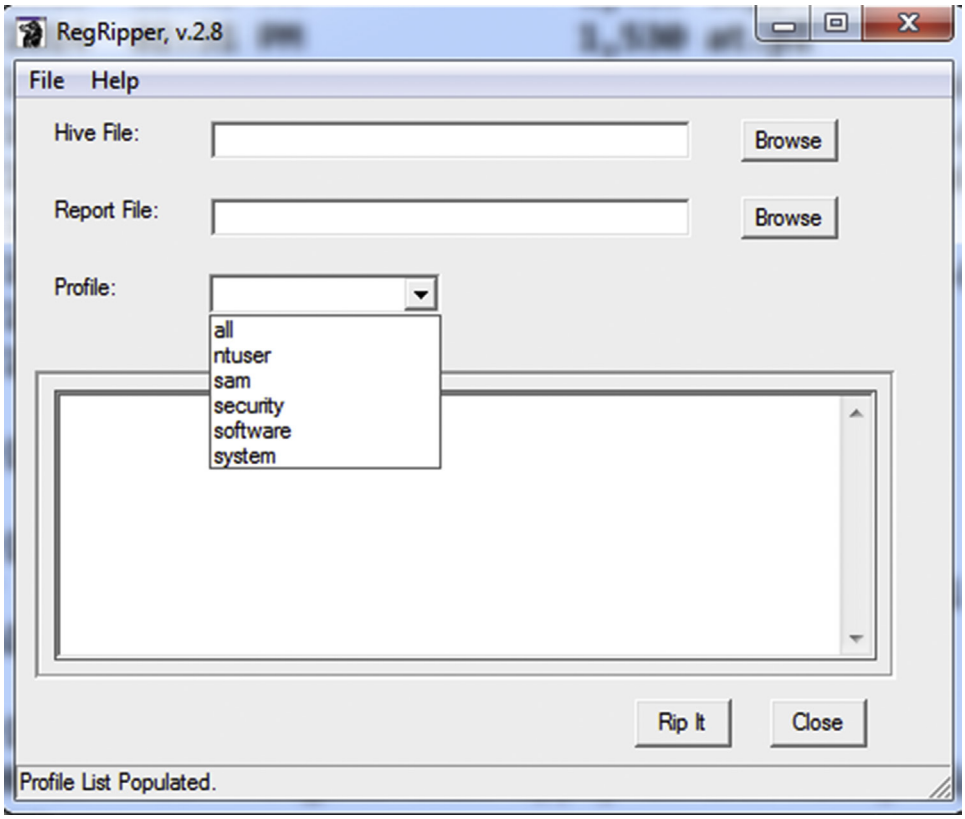

Figure 5.3 RegRipper UI showing available profiles.

Getting the Most Out of RegRipper

I know that to many users, RegRipper seems like a pretty static tool; most users download the tool, run it using GUI, and generally don't do a great deal to really push RegRipper beyond the tool that they downloaded. However, RegRipper is actually pretty flexible and, when used in conjunction with other tools (see chapter Processes and Tools) and analysis processes, can also be a very dynamic framework.

Finding Out About Plugins

Finding out about the plugins that you have available to you is a relatively straightforward affair. One way to do this is to simply open Windows Explorer, click through the directory structure until you get to where the plugins are located, and open them up one at a time in Notepad.

Note

I've noticed over the years a certain reticence on the part of many RegRipper users to take on the responsibility of discovery themselves. I find this odd, because many of these users are in positions where they have to have a certain modicum of confidence; they regularly interact with senior management, interview suspects, or testify in court. I will hear people ask (or, hear that they asked) about a particular plugin, and when I suggest that they open the plugin file in Notepad, I immediately get, "I don't program" or "I don't understand Perl code." You don't have to. Many programming languages provide the ability to add comments to code; these are statements included in the code that are not executed and can be used to provide context or clarity to sections of code. I tend to make use of comments in the plugins I've written, so that if you have questions about a plugin, you can open it in Notepad without fear of changing it (if all you're going to do is read it) to get a little better idea of what the code is doing.

If you do open a plugin in Notepad or Notepad++, and when you go to close it, are asked if you want to save any changes (perhaps due to an inadvertent key stroke), simply choose to close the file without saving the current contents.

I built a capability into rip (RegRipper's little brother) that allows users to get a quick listing of the available plugins. In order to take advantage of this capability, open a command prompt and navigate to the directory where you copied RegRipper and type the following command:

```
C:\perl\rr>rip -l
```

In this case, we're using the "compiled" Windows executable for rip.exe, rather than the Perl script (rip.pl) to run the command. I'm assuming that you downloaded RegRipper, but do not have Perl installed, which is perfectly acceptable.

An excerpt of the output of the command appears as follows:

```
265. typedurls v.20080324 [NTUSER.DAT]
 - Returns contents of user's TypedURLs key.
266. typedurlstime v.20120613 [NTUSER.DAT]
 - Returns contents of user's TypedURLsTime key.
267. typedurlstime_tln v.20120613 [NTUSER.DAT]
 - Returns contents of Win8 user's TypedURLsTime key (TLN).
```

Again, this is just an excerpt; at the time of this writing, my installation of RegRipper has 315 plugins.

So, what the command does is that it goes to the plugins folder (by default, "plugins"), locates all files that end in ".pl", and extracts information from them. The information that you see displayed (name, version number, hive that the plugin is intended for, and a description) is all embedded within the plugin file itself. Of course, there may be more extensive information available in the headers or comments of the plugin file, which you can view by opening the plugin file in Notepad or Notepad++.

Another way to run the same command is to add the "-c" switch, which when used with the "-l" switch (for listing the plugins) will format the output as comma-separated values (CSV), suitable for opening in Excel.

```
C:\perl\rr>rip -l -c
```

You'll notice that the output is displayed on the console. To get this into an Excel file, simply redirect the output to a file as follows:

```
C:\perl\rr>rip -l -c > plugins.csv
```

You can now open the resulting file in Excel, or even Notepad, if you wish. Once you have the output in Excel, you can sort on names, versions, or hive files. Or, you can extend the command line you're using to produce only a list of plugins intended to be run against the NTUSER.DAT hive file by typing:

```
C:\perl\rr>rip -l -c | find "NTUSER" /i > ntuser_plugins.
csv
```

Again, you can open the resulting file in Excel and sort on any of the columns. This is a great way to get a listing of plugins that are run against a particular hive (replace "NTUSER" with any other hive name, or "ALL") so that you can create your own custom profiles, which will be discussed later in this chapter.

Tip

In April, 2013, Adam (aka "Hexacorn") wrote a Perl script that you could run across the plugins that you have available, and it would produce a nice HTML report illustrating the keys that were checked by each plugin. I haven't run it, but I do think that it was a great way to address an issue (or solve a problem) that at least one analyst had encountered. The Perl script (named "3r.pl", for "RegRipper Ripper") can be found online at http://hexacorn.com/tools/3r.pl, and an example of the output of the script, run at the time that the script was created, can be found online at http://hexacorn.com/tools/3r.html.

Creating New Plugins

I promised that you don't have to be a programmer to use or get the most out of RegRipper, and that's still true. However, even without extensive programming skills, you can create your own plugins. As an example, Corey Harrell (author of the "Journey into IR" blog, found online at http://journeyintoir.blogspot.com) created his own plugin by copying the contents of an available plugin into another plugin file, and changing specific items within the

code (ie, Registry key path, value, etc.) to meet his needs. I've used this same "cut-and-paste" approach myself to create more than a few plugins. Let's say that I read something online regarding malware that creates a particular Registry key during installation; knowing the hive file in which the key is located is the first step. I then just open a plugin that extracts data from the same hive file, copy as much of the contents as I need, paste that into another file, and start making the necessary changes for the plugin I want to produce.

Since I first released RegRipper, my request has always been that if someone has a question about RegRipper or a plugin, ask it. Similarly, if someone needs a plugin modified, or a new one created, the best approach is to contact me with a concise description of what you're looking for (maybe include some examples) and provide sample data for testing. In some cases, when someone has asked for assistance in that manner, I've been able to turn around new or updated plugins in 1–4 hours. I have had people ask me for new plugins but be either unwilling or unable to provide sample data, and I really don't want to send someone a plugin that I haven't had an opportunity to test. My goal is to provide a plugin that they can use, not something that the only response is that "it doesn't work."

To my knowledge, there are no "hidden" repositories of RegRipper plugins. For the foreseeable future, the RegRipper distribution will continue to be located online at https://github.com/keydet89/RegRipper2.8. Over the years, a few analysts have provided plugins of their own, and others have requested updates to plugins, as well as providing the necessary data to develop and test the plugin. The most notable case for the latter is the *ares.pl* plugin; I don't have many opportunities to work cases involving peer-to-peer (P2P) file sharing applications, and a district attorney shared an NTUSER.DAT file with me along with a request to update the *ares.pl* plugin in order to address the data that they'd seen.

Tip

I understand that sharing data from cases can be hard to do; however, the instances where someone has shared hive files from actual cases have predominantly been from law enforcement. Not long ago, Cindy Murphy, a detective from Wisconsin who is extremely well known for her extensive knowledge of mobile device forensic analysis, shared Registry hive files from a Windows Phone 8 with me, and from these I was able to determine that many of the current RegRipper plugins worked just fine on these hives. In another instance, a district attorney shared a hive file with me that allowed me to update a plugin.

Create Your Own Profiles

Both the RegRipper GUI tool (rr.exe) and the command line tool (rip.exe) make use of profiles, which are simply files that contain a list of plugins to run. A profile file has no extension; it does *NOT* end in ".txt" nor any other extension.

An easy way to see which profiles you have available is to open a command prompt and navigate to the folder that contains the plugins and type the following command:

```
C:\Perl\rr\plugins>dir /os | more
```

What this command does is list the files within the folder, using a sort order based on the size of the files, listing the smallest files first. This is an effective means of listing the profiles, as you'll see, because the profiles are much smaller than the plugins. An excerpt of the output from the above command, run on the system I use to develop RegRipper, appears as follows:

```
03/29/2013 02:43 PM 77 sam
03/29/2013 02:43 PM 93 usrclass
03/29/2013 02:43 PM 104 security
08/07/2014 04:27 PM 107 all
04/03/2013 09:55 AM 568 system
07/16/2013 06:44 AM 660 software
09/05/2013 07:15 AM 1,259 ntuser
08/21/2014 02:34 PM 1,351 at_tln.pl
07/13/2010 12:45 PM 1,465 skype.pl
```

Notice that the smallest file is "sam", which is only 77 bytes in size. Using the "type" command, you can send the contents of the file to the console, and you'll see what appears as follows:

```
# 20120528 *ALL* Plugins that apply on SAM hive,
alphabetical order
samparse
```

As you can see, there really isn't much to the "sam" profile, as it points to only one plugin, samparse.pl. In fact, the vast majority of the 77 bytes of the file are consumed by the visible comment. That's right, lines beginning with "#" are comments and as such are not processed as commands by RegRipper.

Similarly, the "all" profile contains the following lines:

```
# 20120528 *ALL* Plugins that apply on any HIVES,
alphabetical order
baseline
findexes
regtime
rlo
del
```

The "all" profile contains plugins that can be run against any hive. I know…the file name isn't entirely imaginative, but it is deceptively descriptive.

Once you've created your new profile, be sure to test it by running it against whatever hives you have available. You can do this by closing and relaunching the RegRipper UI and selecting the new profile from the drop-down list. An alternative to this approach is to use rip.exe to run the profile against a hive file using a command similar to the following:

```
C:\perl\rr>rip -r d:\cases\local\ntuser.dat -f new_ntuser_
profile
```

If you want to see all of the output from the above command, be sure to redirect the output to a file, using the appropriate redirection operator; either ">" to create a new file or ">>" to append the information to a new file.

Tip

In chapter Processes and Tools, we discussed various tools you could use, and we discussed those tools as they pertained to different analysis processes. One of the tools we discussed for diff'ing two hive files—displaying the differences between two hives—was a script that shipped as part of the Parse::Win32Registry Perl module. An alternative means to doing the same sort of analysis, albeit on a smaller scale, would be run create a profile, run that same profile against two hive files, and then use native text-based differencing tools such as *fc.exe* or *comp.exe* to compare the output files.

Extending RegRipper

When it comes to getting the most out of RegRipper, another approach to consider is what Corey Harrell came up with in creating "auto_rip", which can be found online at http://journeyintoir.blogspot.com/2013/05/unleashing-autorip.html. Note that at the end of the blog post, Corey provides a download location for the auto_rip script, which he describes as a wrapper script for using RegRipper.

Corey has long been a proponent for "artifact categories," which is a method for classifying various artifacts based on their applicability to various aspects of analysis. For example, if your analysis goals are to determine, as best you could, all of the programs that had been run on a computer system, or all of the applications that had been launched by a specific user, there are a number of data sources and artifacts within a Windows system where you can focus your analysis. The "give me everything, including the kitchen sink" approach to analysis can quickly become overwhelming due

to the sheer volume of data, and the concept of "artifact categories" allows an analyst to focus on specific data sources and artifacts that will provide the analyst with information pertinent to that analysis. As we saw in chapters Analyzing the System Hives and Case Studies: User Hives, there are several keys and values within the Registry that allow an analyst to focus on specific information about which applications were run on a system, which falls under the "program execution" category. Sometimes, various categories can be very similar; for example, there are "program execution" artifacts that will provide information about applications that a user launched, as well as "autostart" artifacts that will provide information about applications that were automatically launched when the user took a specific action, such as logging into the system, logging out, or running an application.

Corey developed auto_rip to be a means for using RegRipper to get information specific to various artifact categories. His blog post lists the various supported categories and even includes the ability to run RegRipper across multiple hive files and retrieve information that pertains to all of the supported categories. This can be extremely valuable to an analyst who has a very clear picture in their mind regarding what their analysis goals are and what questions they are trying to answer. Corey's efforts have extended the ability to use RegRipper not in a single step but rather in a quantum leap.

What to Do When Something Goes Wrong

As with any software, there is the possibility that something will not go as planned or expected. If you get an error message or some other message that you didn't expect to see when you ran RegRipper, there are a couple of things you can check.

First, check to ensure that you ran the plugin against the appropriate hive file. The plugins are written in Perl, which means that for the most part, they're text files and can be opened in an editor, such as Notepad (native to Windows installations) or Notepad++ (available online from http://notepad-plus-plus.org/). Near the beginning of each plugin, there's a Perl hash structure named "%config" that contains metadata for the plugin; within this structure is a value named "hive" that indicates for which hive (or hives, as the case may be) the plugin is intended. This value is illustrated in Fig. 5.4.

Second, check the hive file itself; is it a hive file? Open the hive file in a hex editor and check to see if the first 4 bytes are "regf", and go to offset 4096 (0x1000 in hexadecimal) and check to see if the first 4 bytes at that offset are "hbin". Most importantly, check to ensure that the hive file itself is not full of zeros.

```
package shellbags;
use strict;
use Time::Local;

my %config = (hive          => "USRCLASS\.DAT",
              hivemask      => 32,
              output        => "report",
              category      => "User Activity",
              osmask        => 20, #Vista, Win7/Win2008R2
```

Figure 5.4 Plugin metadata.

If everything looks good and checks out at this point, check the log file. Remember earlier in this chapter (see Fig. 5.2), we discussed the fact that when you run the RegRipper GUI, it creates a log file of activity? Whichever file name you chose for the output file (the file should end with the ".txt" extension) a log file with same name, albeit with the ".log" file extension, will be created in the same folder. If you run into issues with the output of RegRipper, be sure to check this file out. Open it in Notepad or Notepad++ and scroll through it. Do any of the plugins report errors?

Here's an example of an error I found in a log file:

```
  Tue Mar 3 16:19:06 2015: Error in compatassist:
  PLEASE SEE THE PERL2EXE USER MANUAL UNDER "Can't locate
somemodule.pm in @INC"
  FOR AN EXPLANATION OF THE FOLLOWING MESSAGE:
  Can't locate plugins\compatassist.pl in @INC (@INC
contains: PERL2EXE_STORAGE
  C:\Perl\rr C:\Users\harlan\AppData\Local\Temp/p2xtmp-1900)
at C:\Perl\rr\rr.exe line 274.
```

So what does all this mean? Well, this one is pretty easy. See the part where it says "Can't locate plugins\compatassist.pl…?" If I navigate to the plugins folder (via a command prompt) and type "dir compatassist.pl", I get "File not found". Basically, what this error means is that a plugin was included in the profile, but the plugin doesn't exist in the plugins folder. This one is an easy fix…simply open the profile in Notepad and remove the reference to the plugin.

If you still can't figure out what's wrong, feel free to contact me. My e-mail address is included in the header of the plugins that I have written (which is most of them), so you should have no trouble reaching me. I tend to discourage folks from publishing their questions and concerns to random websites, for the simple fact that it would be highly unlikely that I would see them. Whatever

information you can share (beyond, "it doesn't work") about the version of the operating system you were using (as well as the one from which the Registry hive files were extracted), the hive files themselves, the plugins you were running, etc., will only serve to provide me with a better view into what the issue might be and allow me to provide an answer, and possibly even a solution, in a more timely manner.

Summary

RegRipper can be an extremely useful and valuable tool in an analyst's toolkit. RegRipper provides capabilities not available in viewer applications, allowing analysts to parse binary data structures and decode encoded values (or value names). RegRipper can also be used to translate binary data, allowing analysts to parse a globally unique identifier (GUID) out of binary value data and provide for a lookup of what that GUID refers to, displaying it in a human-readable manner.

However, the real power of RegRipper comes from the community of analysts who use and extend the tool, by writing tools such as auto_rip, or by writing their own plugins, and then sharing those tools and plugins with other analysts. To many, the Registry seems like a dark, foreboding abyss, but by working together and sharing knowledge through frameworks such as RegRipper, we can pierce that veil of mystery.