REMnux

By Russ McRee – ISSA member, Puget Sound (Seattle), USA Chapter

Prerequisites

Virtualization platform – VMWare, VirtualBox, etc.

One skill I believe is a core requirement for a well-rounded incident response team is malware analysis. Perhaps you or your associates may not possess deep reverse engineering talents, but I firmly believe you should know how to use certain tools of the trade and interpret the results effectively.

REMnux is intended to provide a collection of tools that can get you started, to both aid you as you learn and enhance your malware analysis capability, or as part of your regular toolkit.

Lenny Zeltser developed REMnux for his SANS Reverse-Engineering Malware course[1] where he utilizes REMnux as he teaches malware analysis.

As I queried Lenny for this article, he indicated that he finds REMnux particularly useful for analyzing malicious software in two primary use-cases.

First, the malware analyst infects a Windows system in a lab and uses REMnux to run services that malware is looking for, such as HTTP, SMTP, and IRC, etc. To meet this requirement, REMnux includes tools such as TinyHTTPd, fakesmtp and Inspire IRCd, as well as Wireshark, fakedns, and honeyd.

The second use-case includes using REMnux to directly analyze malware in the form of malicious browser scripts, Flash and PDF files, and more. For this, REMnux includes a customized version of SpiderMonkey, jsunpack-n, Origami framework, Didier Steven's PDF tools, and Flash decompilers.

Lenny prefers reverse-engineering Windows executables on Windows hosts rather than *nix, which is why REMnux doesn't include Wine. You've read endless *toolsmith* columns where I've made use of a compromised Windows XP virtual machine as I share Lenny's preferences. However, REMnux includes GDB, objdump and Radare to help with shellcode analysis.

Finally, REMnux also includes tools useful for memory forensics such the Volatility Framework and some excellent malware-related plug-ins for it.

Lenny considers REMnux a work in progress and is maintaining a "to-do" list that includes additional tools he'd like to include based on feedback from REMnux users.

He hopes to have an update released in late fall of 2010.

REMnux is meant to be lightweight to allow it to run successfully on hardware that may be a bit dated or that doesn't include a lot of RAM. To that end it uses Enlightenment as the X window manager, rather than GNOME or KDE. REMnux is built on Ubuntu so you can add any tool you wish using apt-get.

If you wish to incorporate the GNOME desktop, ensure that REMnux can connect to the Internet; then simply run `sudo apt-get install ubuntu-desktop` with the understanding that 1GB worth of files will be downloaded as a result (nullifying the premise of lightweight ;-)).

I challenge you to make do with the terminal as you use REMnux. You can hone your *nix and malware analysis skills at the same time.

Lightweight also implies that REMnux doesn't include every single malware analysis tool. The goal is to have the tools the community considers most useful, so that people who are just entering the field of malware analysis have a strong starting point for building and customizing their lab.

Don't be discouraged when you log onto REMnux and simply see a command shell. Lenny's goals include the addition of some hints and shortcuts directly into the interface in the next revision.

In the interim I intend to give you some good starting points and you can certainly learn more about the tools installed on REMnux and find additional getting-started hints at the REMnux website.[2]

REMnux configuration

A few immediately useful tips culled from the REMnux site.

You'll likely want to make use of the SSH server as the easiest method for moving samples and results data to and from your REMnux VM. The commands *sshd start* and *sshd stop* will work as expected, but you may need to generate missing keys if you built from the ISO rather than use the available VM appliance:

```
sudo ssh-keygen -t rsa -f /etc/ssh/ssh_host_
rsa_key -N ''
sudo ssh-keygen -t dsa -f /etc/ssh/ssh_host_
dsa_key -N ''
```

REMnux will automatically pull an IP address as a DHCP client; ifconfig or myip will reveal your address or reacquire your network configuration; use *restart-network*.

I've taken to using Oracle's VirtualBox for quick, straightforward virtualization. As such I downloaded the REMnux

1 http://LearnREM.com.

2 http://REMnux.org.

©2010 Information Systems Security Association • www.issa.org • editor@issa.org • Permission for author use only.

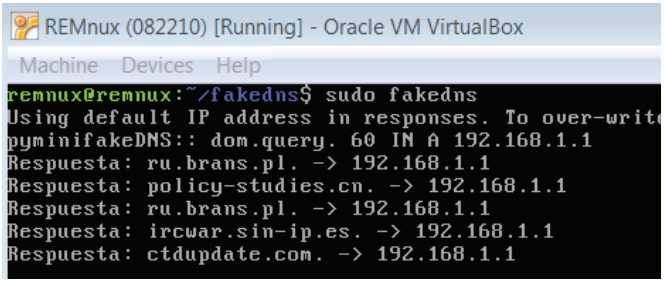

Figure 1 – fakedns answering for Storm, Virut, and Zeus

ISO, and booted from same, then took a preliminary snapshot. Remember to take another one after you've tweaked REMnux to your liking. If you're using VMWare, simply download the appliance and you're good to go.

If you wish to work from Enlightenment (X windows manager), simply execute *startx* after the VM is fully initialized.

REMnux for services

When you really don't want malware to phone home via your Internet provider, but you want it to believe it has every capability to do so, fakedns is useful for trapping DNS lookups. A "minimal Python DNS server," running fakedns is as simple as running `sudo fakedns`. Remember to point your Windows-based malware analysis VM DNS setting to the REMnux IP address.

If you'd like to interact with IRC-based malware, utilize Inspire IRCd via *ircd start*.

Your best bet is the really slick little services simulation suite found in INetSim. You'll have to edit the config file before it will play nicely, but here's how. Remember if you choose to use INetSim for services, kill fakedns so there is no port contention for 53.

Execute `sudo vi /etc/inetsim/inetsim.conf` and modify `service_bind-address` to the match the IP address your VM is configured with. Repeat this step for `dns_default_ip`.

Thereafter, run `sudo inetsim` and let loose some badness on your Window malware VM (remember to point DNS accordingly).

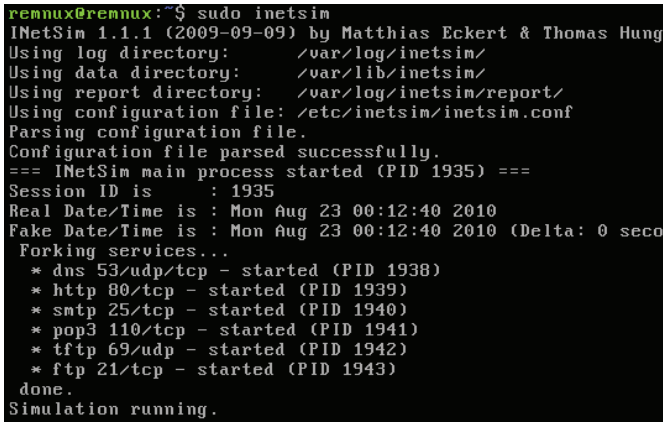

Figure 2 – INetSim initialized

INetSim writes to `/var/log/inetsim` by default including `/var/log/inetsim/service.log` for service activity captured. Figure 2 shows a brief INetSim log snippet captured after I stomped on a variety of malicious binaries on a Windows VM configured as 192.168.248.113.

Note the entries in Figure 3 where 192.168.248.113 makes a DNS request for *www.partizangroup.net* then shortly thereafter phones home via port 80. A quick search engine query for partizangroup.net reveals that it was an active malware domain in 2008, added to the DNS-BH (Black Hole DNS Sinkhole).

```
[2010-08-23 00:34:29] [2063] [dns 53/udp/tcp 2066] [192.168.248.113] send: www.p
artizangroup.net 3600 IN A 192.168.248.110
[2010-08-23 00:34:29] [2063] [dns 53/udp/tcp 2066] [192.168.248.113] disconnect
[2010-08-23 00:34:29] [2063] [dns 53/udp/tcp 2066] [192.168.248.113] stat: 1 qty
pe=A qclass=IN qname=www.partizangroup.net
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] connect
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: POS
T /newlester.php HTTP/1.0
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: Con
nection: keep-alive
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: Con
tent-Type: application/x-www-form-urlencoded
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: Con
tent-Length: 283
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: Hos
t: www.partizangroup.net
[2010-08-23 00:34:29] [2063] [http 80/tcp 2084] [192.168.248.113:1287] recv: Acc
ept: text/html, */*
```

Figure 3 – INetSim services intercepting infected host traffic

INetSim writes HTTP POST data to `/var/lib/inetsim/http/postdata/`. After running `sudo cat /var/lib/inetsim/http/postdata/7db070cbb57a0a814a67caf0132ef4299cd30032 > inetsimPOST.txt` as seen in Figure 3, I grabbed the output and dropped it to a URL decoder.

What was

```
praquem=nadielepf%40terra%2Ecom%2Ebr&titulo=HIO
%2D66ZKDGUCPVW+Foi+infectado+%3AD&texto=Computa
dor+%2E%2E%2E%2E%3A+HIO%2D66ZKDGUCPVW+00%2D0C%2D
29%2D56%2D54%2D89%0D%0AData+%2E%2E%2E%2E%2E%2E%2E%2E
%2E%2E%2E%3A+8%2F22%2F2010%0D%0AHora+%2E%2E%2E%2E
%2E%2E%2E%2E%2E%2E%2E%3A+9%3A34%3A30+PM%0D%0A&
```

became

```
praquem=nadielepf@terra.com.br&titulo=HIO-
66ZKDGUCPVW Foi infectado :D&texto=Computador
....: HIO-66ZKDGUCPVW 00-0C-29-56-54-89
Data .....: 8/22/2010
Hora .....: 9:34:30 PM
```

What would have been an email phone home to the bot herder who owned nadielepf@terra.com.br a couple of years ago was neatly captured by INetSim.

REMnux for direct analysis

One of my teammates recently handled an incident that initially looked like a DoS attack as the requests for the service where excessive to the point of causing latency and service degradation.

The GET request was clearly not known-good for one of our services and weirder still the referrer was an URL identifying a GIF file.

```
GET /fcg-bin/cgi_emotion_list.
fcg?uin=454682113&loginuin=&s=&.swf HTTP/1.1
```

©2010 Information Systems Security Association • www.issa.org • editor@issa.org • Permission for author use only.

```
Accept: */*
Accept-Language: zh-CN
Referer: http://img.qqywf.com/logo.gif
```

Downloading logo.gif to my REMnux VM presented an excellent opportunity to test certain static executable and binary analysis tools included with REMnux.

First, as I suspected that logo.gif was not actually a GIF, I used TrID,[3] a utility designed to identify file types from their binary signatures.

Executing trid logo.gif yielded Figure 4.

```
remnux@remnux:~/malware/flash$ trid logo.gif

TrID/32 - File Identifier v2.00/Linux - (C) 2
Definitions found:  3887
Analyzing...

Collecting data from file: logo.gif
100.0% (.SWF) Macromedia Flash Player
 Compressed Movie (3002/2)
```

Figure 4 – TrID identifies a disguised Flash file

Ah, so having confirmed a Flash file, I moved to the Flash tools available, including swftools, flasm, and flare. I ran `flasm logo.swf` after renaming logo.gif to logo.swf. Flasm disassembles SWF files, including all the timelines and events. **Note**: Keep in mind that flasm doesn't support ActionScript 3.1. Lenny only included flasm because of its convenient `flasm -x` feature to decompress swf files. For disassembling files, it's better use `swfdump -Ddu` instead.

The results are written to an output file; in this case logo.flm. Scrolling through the results via cat, I found: `http://g.qzone.qq.com/fcg-bin/cgi_emotion_list.fcg?uin=454682113&loginuin=&s=&.swf`

This closed the loop on what was happening. The initial GET request for our service included `/fcg-bin/cgi_emotion_list.fcg?uin=454682113&loginuin=&s=&.swf` as noted above but was obviously resulting in a 404 error as the request resource was located on http://g.qzone.qq.com. Misconfiguration or intentional? Hard to say, but we black holed it given the impact to the service.

Malicious PDFs

Malicious PDF analysis is another strong suit of the tool suite included with REMnux.

You will find Jsunpack-n and Didier Stevens' PDF tools[4] mentioned earlier useful in this regard.

I grabbed a malicious PDF sample from offensivecomputing.net (`MD5: a491ae05103849d8797d1fda034e0bd5, readme.pdf`) and put it to use on REMnux as follows.

Didier recommends using pdfid.py first to identify a "given list of strings and count the occurrences (total and obfuscated) of each word." Jsunpack-n's pdf.py script would have provided similar functionality.

This list includes /JS and /JavaScript which, when identified in a PDF sample, most often indicate maliciousness. pdfid.py readme.pdf resulted in Figure 5.

Figure 5 – PDFid identifies keywords

readme.log

```
1  PDFiD 0.0.10 readme.pdf
2   PDF Header: %PDF-1.1
3   obj                    10
4   endobj                 10
5   stream                  3
6   endstream               3
7   xref                    1
8   trailer                 1
9   startxref               1
10  /Page                   1
11  /Encrypt                0
12  /ObjStm                 0
13  /JS                     1
14  /JavaScript             1
15  /AA                     0
16  /OpenAction             1
17  /AcroForm               0
```

With a clear indicator of JavaScript inclusion, you can then use `pdf-parser.py` to learn further details.

The last handy little script we'll cover here (there are many more for you to discover and test) is Jim Clausing's packerid.py. Jim indicates that while he likes PEiD,[5] it's Windows-only. He wanted a script which uses a PEiD database to identify which packer (if any) is being used by a binary that runs on a *nix OS, so he wrote it for himself.

It's as easy as running `packerid malware.exe`.

The output will either be *None* or as seen in the following examples:

```
['ASPack v2.12 -> Alexey Solodovnikov'](MD5:
f7628666adcb35491a925f240f97c634)
['ASProtect v1.23 RC1'] (MD5: 01bcd-
0d218157ce6c0f596676864833c)
```

Experiment with bytehist too; it helps you visualize statistical data to detect encrypted or packed data.

In conclusion

If you're looking to develop or accentuate your malware analysis skills, REMnux is a great resource. Keep an eye on the project site for the next release, and by all means send feedback to Lenny. Contact him via lennyzeltser on Twitter and his website at zeltser.com. If the suggestions are useful, they will be incorporated as appropriate.

There are a number of tools found on REMnux, so don't hesitate to dig in. Read the project web site along with the reference URLs it provides for each tool; your efforts will be rewarded.

Cheers…until next month.

Acknowledgements

Lenny Zeltser, for REMnux, and championing the cause of teaching malware analysis.

About the Author

Russ McRee, GCIH, GCFA, GPEN, CISSP, is team leader and senior security analyst for Microsoft's Online Services Security Incident Management team. As an advocate of a holistic approach to information security, Russ' website is holisticinfosec.org. Contact him at russ@holisticinfosec.org.

3 http://mark0.net/soft-trid-e.html.

4 http://blog.didierstevens.com/programs/pdf-tools.

5 http://peid.has.it.

©2010 Information Systems Security Association • www.issa.org • editor@issa.org • Permission for author use only.