Violent Python: A Book Review Applied to Security Analytics

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

Prerequisites/dependencies

Python interpreter

BackTrack 5 R3 is ideally suited to make immediate use of Violent Python scripts

Happy New Year and congratulations on surviving the end of the world as we know it (nyah, nyah Mayan calendar). Hard to imagine we're starting yet another year already; 2012 simply screamed by. Be sure to visit the HolisticInfoSec blog post[1] for the 2012 Toolsmith Tool of the Year and vote for your favorite tool of 2012.

I thought I'd start off 2013 with a bit of a departure from the norm. Herein is the first treatment of a book as a tool where the content and associated code can be utilized to perform duties specific to the information security practitioner. I can think of no better book with which to initiate this approach than TJ O'Connor's *Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers, and Security Engineers.* Yes, this implies that you should buy the book; trust me, it's worth every dime. Better still, TJ has donated all his proceeds to the Wounded Warrior Project.[2] That said, I'll post TJ's three scripts we'll discuss here so as to whet your appetite. I've had the distinct pleasure of working with TJ as part of the SANS Technical Institute's graduate program where we, along with Beth Binde, wrote *Assessing Outbound Traffic to Uncover Advanced Persistent Threat.*[3] I've known some extremely bright capable information security experts in my day and I can comfortably say TJ is hands down amongst the very best of that small group. As part of his service as an officer in the US Army (hooah) TJ has served as the course director for both computer exploitation and digital forensics at the US Military Academy and as an communications officer supporting tactical communications. His book maps nicely to a philosophy I embrace and incorporate in the workplace. Security monitoring, incident response (and forensics), and attack and penetration testing are the three pillars of security analytics, each feeding and contributing the others in close cooperation. As an example, capable security monitoring inevitably leads to a need for incident response, and after mitigation and remediation have ensued, penetration testing is key to validating that corrective measures were successful, which in turn helps the monitoring team assess and tune detection and alerting logic. Security analytics: the information security circle of life ☺.

How does a book such as TJ's *Violent Python* reverberate with this philosophy? How about entire chapters dedicated to each of the above mentioned pillars, including Python scripts for network traffic analysis (monitoring), forensic investigations (IR), as well as web recon and penetration testing. We'll explore one script from each discipline shortly, but not before hearing directly from the author:

"In a lot of ways writing a book is a cathartic experience where you capture a lot of things you have done. All too often I'm writing scripts to achieve an immediate effect and then I throw away the script. For me the book was an opportunity to capture a lot of those small projects I've done and simplify the learning curve for others. My favorite example was the UAV takeover in the book. We show how to take over any really ad hoc WiFi toys in under 70 lines of code. A few friends joked that I couldn't write a script in under 100 lines to crash a UAV. This was my chance to provide them a working concept and it worked! Unfortunately it left my daughter with a toy UAV cracked into several pieces as I refined the code. From a defensive standpoint, understanding a scripting language is absolutely essential in my opinion. The ability to parse data such as DNS traffic or geo-locate IP traffic (both shown in the book) can give a great deal of visibility. Forensics tools are great but the ability to build your own is even better. We show how to write tools to parse out iPhone backups for data and scrape for specific objects. The initial feedback from the book has been overwhelming and I've really enjoyed hearing positive feedback. No future plans right now but a good friend of mine has mentioned writing "Violent Powershell," so we'll see where that goes."

Violent Python provides readers the basis for scripts to attack network services, analyze digital artifacts, investigate network traffic for malicious activity, and data-mine social media, not to mention numerous other activities. This is a must-read book that includes a companion site with all the code discussed. Let's take a closer look at three of these efficient and useful Python scripts.

1 http://holisticinfosec.blogspot.com/2012/12/choose-2012-toolsmith-tool-of-year.html.

2 http://www.woundedwarriorproject.org.

3 http://www.sans.edu/student-files/projects/JWP-Binde-McRee-OConnor.pdf.

Making Use of Violent Python

As noted above, I've posted the three scripts discussed in this section, along with the PCAP and PDF (malicious) discussed on my website. Email or Tweet for the zip passwords.

TJ suggests utilizing a BackTrack distribution, given that many of the dependencies and libraries required to use the scripts in this book are inherent to BackTrack. We'll follow suit on a BackTrack 5 R3 virtual machine. Before beginning, we'll need to set up a few prerequisites. Execute `easy_install pyPDF python-nmap pygeoip mechanize BeautifulSoup4` at the *BT5R3 root* prompt. This will install pygeoip as needed for our first exercise. I'm going to conduct these exercises a bit out of chapter sequence in order to follow the security analytics life cycle starting with monitoring. This drops us first into Chapter 4 where we'll utilize MaxMind's GeoLiteCity to map IP addresses to cities. In order to do so, you'll need to set up GeoLiteCity on BackTrack or your preferred system with the following steps:

1. `mkdir /opt/GeoIP`
2. `cd /opt/GeoIP/`
3. `wget http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz`
4. `gunzip GeoLiteCity.dat.gz`

You'll then need to edit line 7 of `geoPrint.py` to read `gi = pygeoip.GeoIP('/opt/GeoIP/GeoLiteCity.dat')` or download the updated copy of the script I've posted for you.

I've created a partially arbitrary scenario for you with which to walk through the security analytics life cycle using *Violent Python*. To do so I'll refer to what was, in 2009, an actual malicious domain, used to host shellcode for PDF-based malware attacks. I grabbed a malicious PDF sample from Contagio, an excellent sample resource. The IP address I associate with this domain is where I am taking creative liberties as the domain we'll discuss, ax19.cn, no longer exists, and there is no record of what its IP address was when it was in use. The PCAP we'll use here is one I edited with bittwiste to arbitrarily introduce a suspect Chinese IP address to what was originally a packet capture from a machine compromised by Win32.Banload.MC. I've shared this PCAP and the PDF as mentioned above so you can try the Python scripts with them for yourself.

In this scenario, your analysis machine is Linux only. Just you, a Python interpreter, and a shell; no fuss, no muss.

As we're starting in the monitoring phase, imagine you have a network for which the traffic baseline is well understood. You can assert, from one particular high value VLAN, that at no time should you ever see traffic bound for China. Your netflow monitoring for that VLAN is showing far more egress traffic bound for IP space that is not on your approved list established from learned baselines. You initiate a real-time packet capture to confirm. Capture (suspect.pcap) in hand, you'd like to validate that the host is indeed conversing with an IP address in China. *Violent Python's* geoPrint.py script is a great place to start as it leverages the above-mentioned GeoLiteCity data from MaxMind along with the PyGeoIP library from Jennifer Ennis and dpkt. Execute python geoPrint.py `-p suspect.pcap` and you'll see results as noted in figure 1.

```
[+] Src: 116.254.188.24 --> Dst: 192.168.248.114
[+] Src: Beijing, CHN--> Dst: Unregistered
[+] Src: 192.168.248.114 --> Dst: 116.254.188.24
[+] Src: Unregistered--> Dst: Beijing, CHN
[+] Src: 116.254.188.24 --> Dst: 192.168.248.114
[+] Src: Beijing, CHN--> Dst: Unregistered
[+] Src: 192.168.248.114 --> Dst: 116.254.188.24
[+] Src: Unregistered--> Dst: Beijing, CHN
[+] Src: 192.168.248.114 --> Dst: 116.254.188.24
[+] Src: Unregistered--> Dst: Beijing, CHN
[+] Src: 192.168.248.114 --> Dst: 116.254.188.24
[+] Src: Unregistered--> Dst: Beijing, CHN
[+] Src: 116.254.188.24 --> Dst: 192.168.248.114
[+] Src: Beijing, CHN--> Dst: Unregistered
[+] Src: 116.254.188.24 --> Dst: 192.168.248.114
[+] Src: Beijing, CHN--> Dst: Unregistered
root@bt:~/ViolentPython/CH4#
```

Figure 1 – geoPrint.py confirms Chinese takeout

Your internal host (RFC 1918, and thus unregistered) with IP address `192.168.248.114` is clearly conversing with `116.254.188.24` in Beijing. Uh-oh.

Your team now moves into incident response mode and seizes the host in question. You interview the system's user who indicates he received an email he thought was a legitimate help desk notification to read a new policy. The email had an attached PDF file which the user downloaded and opened. Your suspicions are heightened, as such you grab a copy of the PDF and head back to your analysis workstation. You're interested to see if there is any interesting metadata in the PDF that might help further your investigation. You refer to Chapter 3 of *Violent Python* which discusses forensic investigations with Python. The pdfRead.py script incorporates the PyPDF library which allows you to extract PDF document information (metadata) in addition to other capabilities. Execute python `pdfRead.py -F suspect.pdf` and dump the metadata as seen in figure 2.

```
root@bt: ~/ViolentPython/CH3
File  Edit  View  Terminal  Help
root@bt:~/ViolentPython/CH3# python pdfRead.py -F suspect.pdf
[*] PDF MetaData For: suspect.pdf
[+] /CreationDate:D:20090506204524+08'00'
[+] /Producer:DocuCom PDF Core Library
[+] /Enhanced:By PDF Enhancer 3.2/Win
[+] /SPDF:1122r
[+] /ModDate:D:20090818195938+08'00'
[+] /Author:Zeon Technical Publications
root@bt:~/ViolentPython/CH3#
```

Figure 2 – pdfRead.py dumps suspect PDF metadata

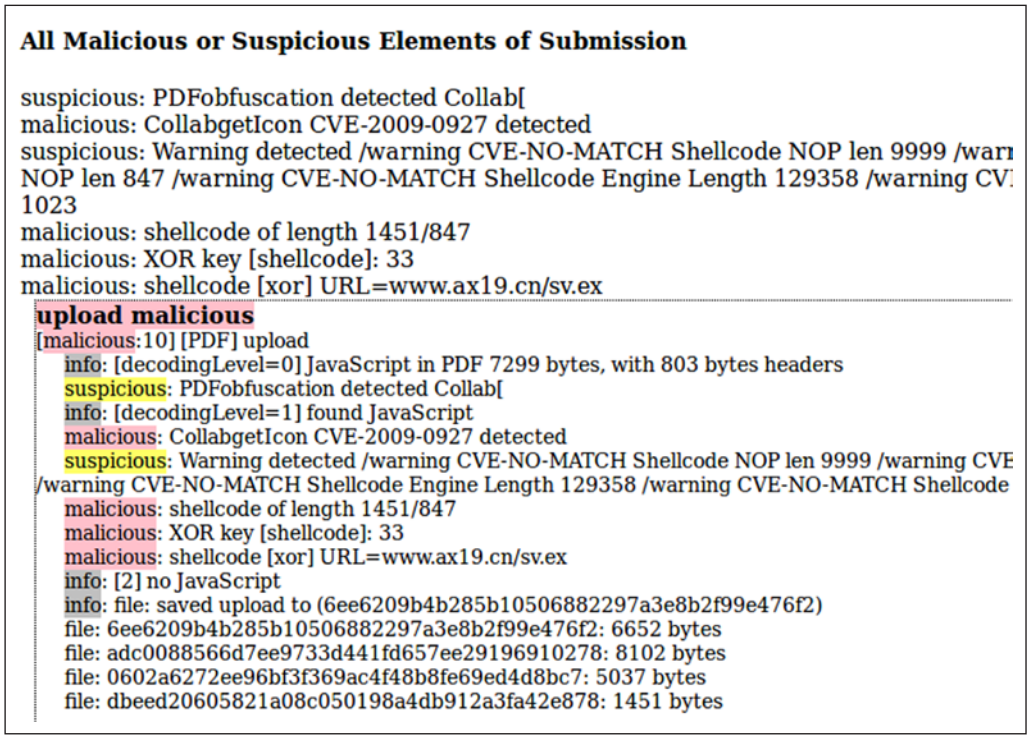

Figure 3 – JSunpack confirms an evil PDF

The author reference is a standout for you; from a workstation with a browser you search "Zeon Technical Publications" and find reference to it on VirusTotal and JSunpack; these results along with a quick MD5sum hash match indicate that this PDF is clearly malicious. The JSunpack reference indicates that shellcode phones home to www.ax19.cn (see figure 3), a domain for which you'd now like to learn more.

You could have sought anonymity to conduct the above mentioned search, which lead us to the third pillar of our security analytics life cycle. This third phase here includes web recon as discussed in Chapter 6 of *Violent Python*, a common step in the attack and penetration testing discipline, to see what more we can learn about this malicious domain. As we often seek anonymity during the recon phase, *Violent Python* allows you maintain a bit of stealth by leveraging the deprecated Google API against which a few queries a day can still be executed. The newer API requires a developer's key which one can easily argue is not anonymous. Executing python `anonGoogle.py -k 'www.ax19.cn'` will return yet another validating result as seen in figure 4.

With seven rich chapters of Python goodness, *Violent Python* represents a golden opportunity to expand your security analytics horizons. There is so much to learn from here while accentuating your use of Python in your information security practice.

In conclusion

I'm hopeful this slightly different approach to *toolsmith* was useful for you this month. I'm looking to shake things up a bit here in 2013 and am certainly open to suggestions you may have regarding ideas and approaches to doing so. *Violent Python* was a great read for me and a pleasure to put to use for both this article as well as in my personal tool box. I'm certain you'll find this book equally useful.

Ping me via email if you have questions (russ at holisticinfosec dot org).

Cheers…until next month.

Acknowledgements

—TJ O'Connor, Violent Python author

—Mila Parkour, Contagio

About the Author

Russ McRee manages the Security Analytics team (security incident management, penetration testing, monitoring) for Microsoft's Online Services Security & Compliance organization. In addition to toolsmith, he's written for numerous other publications, speaks regularly at events such as DEFCON, Black Hat, and RSA, and is a SANS Internet Storm Center handler. As an advocate for a holistic approach to the practice of information assurance Russ maintains holisticinfosec.org. He serves in the Washington State Guard as the Cybersecurity Advisor to the Washington Military Department. Reach him at russ at holisticinfosec dot org or @holisticinfosec.

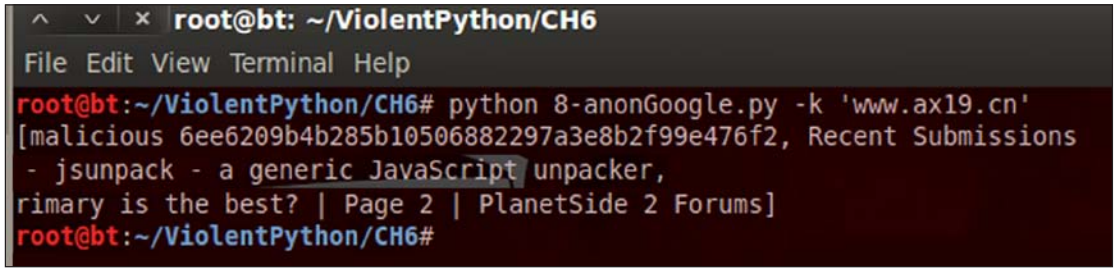

Figure 4 – anonGoogle matches ax19.cn to malicious activity