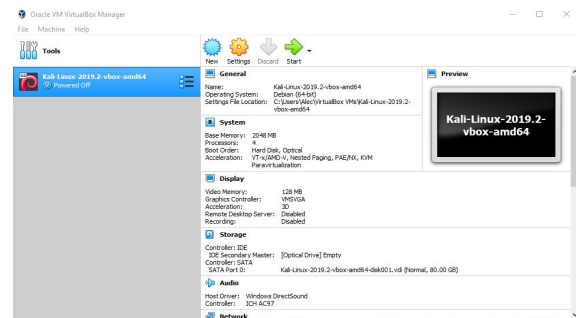


Alec Miller  
July Project

## Recon-NG

Recon-NG is a powerful OSINT penetration tool that I explored for my second project as the Plessas Experts Network intern. While Recon-NG is limited to running in a Linux environment, there are workarounds (dual booting or downloading a virtual machine) that allow a Windows PC user to access its capabilities. For my examples, I used Oracle's virtual machine known as [VirtualBox](#)\* to run Kali-Linux on my PC in order to run Recon-NG. For help with downloading and running the virtual machine and Kali-Linux, [this video](#) is very helpful.



When first launching Recon-NG, expect that many API key errors will appear. This is perfectly fine and will be handled later. Recon-NG has three simple, but very effective tools that are constantly in use with this program: “show,” “help,” and “clear.” [Show](#)\* displays all the usages and what can be viewed using the show command.

```
[recon-ng][default] > show
Shows various framework items

Usage: show [banner|companies|contacts|credentials|dashboard|domains|hosts|keys|leaks|locations|modules|netblocks|options|ports|profiles|pushpins|repositories|schema|vulnerabilities|workspaces]
```

Similarly, when using the [help](#)\* command, the program displays how the command works and what properties go with it. The [clear](#)\* command clears away any text on the screen to keep everything neat. Lastly, the [back](#)\* command removes whatever module is currently set.

```
[recon-ng][default] > help

Commands (type [help?] <topic>):
-----
add           Adds records to the database
back         Exits the current context
delete       Deletes records from the database
exit         Exits the framework
help        Displays this menu
keys        Manages framework API keys
load        Loads specified module
pdb         Starts a Python Debugger session
query       Queries the database
record      Records commands to a resource file
reload      Reloads all modules
resource    Executes commands from a resource file
search     Searches available modules
set         Sets module options
shell       Executes shell commands
show       Shows various framework items
snapshots  Manages workspace snapshots
spool      Spools output to a file
unset      Unsets module options
use        Loads specified module
workspaces Manages workspaces
```

A good starting place in Recon-NG is a new workspace which is displayed with the “show” command. Workspaces are comparable to saved locations in a video game. Any data gathered on that workspace will be saved for later viewing or use. To begin creating your workspace, see the commands for using the workspaces by entering, “help workspaces.” After adding a [new workspace](#)\*, the command line will change to match the new workspace. In my case, the command line now displays “PEN” since that’s what I named my new

```
[recon-ng][PEN] >
```

\*Hyperlinks of images are given for larger versions of screenshots.

workspace. Since a new workspace has been created, there is no information that we have gathered or added in this space. This can be seen after typing “show” and one of the options (for this example, I used “show companies”), it [displays](#)\* that we have no data to be viewed. If a word gets misspelled and the user wants to remove them from the database, the “delete” command allows the user to individually remove items by specifying their rowid, delete multiple selections by using a comma between rowids, and bulk-delete items using a dash between the two netids.

```
[recon-ng][PEN] > show companies  
[*] No data returned.
```

There is some set-up required before beginning to work with Recon-NG’s many modules. This required set-up includes adding the API keys to the program which will address the errors we see when first launching Recon-NG.

```
[recon-ng][PEN] > show keys  
+-----+-----+  
| Name | Value |  
+-----+-----+  
|bing_api|  
|builtwith_api|  
|censysio_id|  
|censysio_secret|  
|flickr_api|  
|fullcontact_api|  
|github_api|  
|google_api|  
|hashes_api|  
|ipinfodb_api|  
|ipstack_api|  
|jigsaw_api|  
|jigsaw_password|  
|jigsaw_username|  
|pwnedlist_api|  
|pwnedlist_iv|  
|pwnedlist_secret|  
|shodan_api|  
|twitter_api|  
|twitter_secret|  
|virustotal_api|  
+-----+-----+
```

Figuring out where to acquire these keys was initially tricky for me. Thankfully, Github user Raikia created a [Recon-NG API key helper](#)! Once the keys have been added, the user can type “[show keys](#)”\* to visually confirm their keys are there. The API key cheat sheet was made for a previous version of Recon-NG, so a few modules are missing from that page. The first missing module is ipstack\_API, but the key can be located on [the ipstack website](#). Click on “get free API key,” fill out the information, and receive the key. The second missing key is the bing\_API. Here is [a tutorial](#) on how to get this key. Finally, pwnedlist is no longer an available service, so there will not be a key for them. These keys are necessary for running some of the most useful modules for OSINT collection such as the ipinfodb API.

Recon-NG has a great variety of modules, including: discovery, exploitation, import, recon, and reporting. As Recon-NG is focused on reconnaissance, this category has the most robust variety of modules while other categories offer just a few choices. [All of the modules](#)\* can be viewed by typing the “show modules” command. These modules only work once their required information has been added. Required information for each module is indicated by the “recon/” prompt. For example, the top four modules in the screenshot to the right each require a host. The message after the second “/” is the type of table that will be delivered based on information gathered by the scan. In this case, the “shodan\_ip” module requires a host and will return ports based on data gathered from its scan.

```
recon/hosts-hosts/ssltools  
recon/hosts-hosts/virustotal  
recon/hosts-locations/migrate_hosts  
recon/hosts-ports/shodan_ip  
recon/locations-locations/geocode
```

While I found modules could be initially confusing, each module has a description that explains its function and type of information necessary to complete the scan. These instructions were displayed when I selected a module and used the “show info” command. With the module

\*Hyperlinks of images are given for larger versions of screenshots.

“shodan\_ip” from the previous photo as our example, I used the “show info” command and then [information](#)\* detailing the module is provided.

```
[recon-ng][PEN] > use shodan ip
[recon-ng][PEN][shodan_ip] > show info

Name: Shodan IP Enumerator
Path: modules/recon/hosts-ports/shodan_ip.py
Author: Tim Tomes (@LaNMaSteR53) and Matt Puckett (@t3lc0)
Keys: shodan_api

Description:
Harvests port information from the Shodan API by using the 'ip' search operator. Updates the 'ports'
table with the results.

Options:
-----
Name      Current Value  Required  Description
-----
LIMIT     1              yes       limit number of api requests per input source (0 = unlimited)
SOURCE    default        yes       source of input (see 'show info' for details)

Source Options:
default   SELECT DISTINCT ip_address FROM hosts WHERE ip_address IS NOT NULL
<string> string representing a single input
<path>   path to a file containing a list of inputs
query <sql> database query returning one column of inputs
```

Once a module has been selected and the required information has also been recorded, it's time to scan and use the program to grab information. To begin using any module, simply type the “run” command. If a case appears such as [this one](#)\*, where there are multiple pieces of information but only one needs to be scanned; the rowid can be used after the “run” command to specify which source to use. In the example, to use the Desert iNET domain, the corresponding run command is “run 2”.

```
[recon-ng][PEN][shodan_ip] > show domains

+-----+
| rowid |      domain      |      module      |
+-----+
| 1     | plessas.net     | migrate_hosts   |
| 2     | www.desertinet.com | user_defined    |
+-----+

[*] 2 rows returned
```

Recon-NG finds a lot of data, so it is important to be aware of the parameters of your search. When I used the google.com domain, it seemed like the module's search would not reach an endpoint. The command “ctrl+C” can be used to stop the module and keep the already gathered information. In my test, I used the domain “gmail.com” followed by the “whois\_pocs” module. Here is a comparison between information received when I used [ctrl+C](#)\* versus [letting the module run](#)\*.

```
[*] [contact] <blank> Abuse (abuse@emgmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE7400-ARIN
[*] [contact] <blank> ABUSE (lrockc@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE7401-ARIN
[*] [contact] <blank> ABUSE (lrockc@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE4920-ARIN
[*] [contact] Abuse Abuse (andrewm013@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE5119-ARIN
[*] [contact] <blank> Abuse Department (17brownj@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE5371-ARIN
[*] [contact] <blank> ABUSE MAILBOX (1wow55886@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE5518-ARIN
[*] [contact] <blank> ABUSE MAILBOX (a2zvoip.solutions@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE5536-ARIN
[*] [contact] <blank> ABUSE MAILBOX (aidacsjarudaz3@gmail.com) - Whois contact
[*] URL: http://whois.arin.net/rest/poc/ABUSE5538-ARIN
^C
-----
SUMMARY
-----
[*] 22 total (18 new) contacts found.
```

Options within modules allow users to customize their search settings. The information tab for the “twitter” module has two different options: RADIUS and SOURCE. For most of my searches, I kept the source as default, but this can be a useful customization. While the default location scan has a radius of 1 kilometer, the command “[set radius 2](#)” adjusts the search to a 2km radius, for example.

\*Hyperlinks of images are given for larger versions of screenshots.

```

recon-ng[[PBN][twitter] > show info
-----
Name: Twitter Geolocation Search
Path: modules/recon/locations-pushpins/twitter.py
Author: Tim Tomes (@LaMMAsteR53)
Keys: twitter_api, twitter_secret

Description:
Searches Twitter for media in the specified proximity to a location.

Options:
-----
Name      Current Value  Required  Description
-----
RADIUS    1               yes       radius in kilometers
SOURCE    default        yes       source of input (see 'show info' for details)

Source Options:
-----
default:  SELECT DISTINCT latitude || ' ' || longitude FROM locations WHERE latitude IS NOT NULL AND longitude IS NOT NULL
--setting: string representing a single input
--paths: path to a file containing a list of inputs
--query: database query returning one column of inputs

recon-ng[[PBN][twitter] > set radius 2
RADIUS => 2

```

```

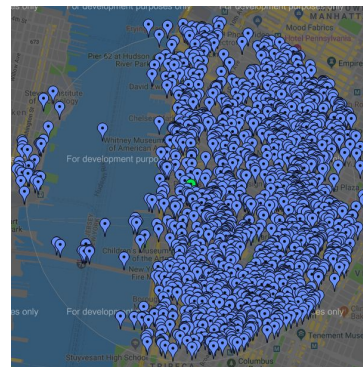
recon-ng[[ME][pushpin] > show info
-----
Name: PushPin Report Generator
Path: modules/reporting/pushpin.py
Author: Tim Tomes (@LaMMAsteR53)

Description:
Creates HTML media and map reports for all of the PushPins stored in the database.

Options:
-----
Name      Current Value  Required  Description
-----
LATITUDE  0               yes       latitude of the epicenter
LONGITUDE 0               yes       longitude of the epicenter
MAP_FILENAME /root/.recon-ng/workspaces/ME/pushpin_map.html  yes       path and filename for pushpin map report
MEDIA_FILENAME /root/.recon-ng/workspaces/ME/pushpin_media.html  yes       path and filename for pushpin media report
RADIUS    1               yes       radius from the epicenter in kilometers

```

In my opinion, one of the most interesting or terrifying discoveries about Recon-NG is the “pushpins” reporting module within its Twitter and Flickr searches. This module uses Google’s API and displays a map with the pinpoint locations of the posts uncovered by previous scans. The [pushpin’s info](#)\* is shown to the right. The latitude, longitude, and radius portions of this module are required for this function. To add this information use the “set” command from before. Example: “set latitude 78.0192”. Once all information has been set, run the module. A Firefox web browser will pop-up with two tabs; one for [the media](#)\* and one for [the map](#)\*. These both have links to every post recorded by Recon-NG. Each pushpin on the map is a different tweet gathered by the Twitter module.



If you are interested in coding, especially in Python, Recon-NG offers a valuable resource. Its “source” tool, paired with the show command is useful to learn from or reference while working with Python. The “show source” command is available to use with any active module. This command brings up the [Python code](#)\* for the active module. The detailed code behind each module fascinating and, by using this reference, anyone is capable of creating more modules for Recon-NG.

Remember, this project paper is only an introduction to Recon-NG’s capabilities. I look forward to using the guidelines and commands described above to begin OSINT investigation and data collection with Recon-NG and continuing to learn more about its other capabilities.

\*Hyperlinks of images are given for larger versions of screenshots.