



Automated PenTest Toolkit

Adam Compton, Senior Security Consultant

(absent) Austin Lane, Security Consultant

Who Am I?

- Adam in 5 words:

- Father *5 years*
- Husband *16 years*
- Hillbilly *39 years*
- Pentester *15+ years*
- Programmer *20+ years*



Overview

- Penetration testing often begins with a simple routine.



Overview

- Penetration testing often begins with a simple routine.
- This routine can be slow on large networks.

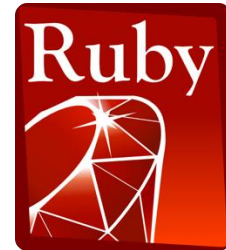


Overview

- Penetration testing often begins with a simple routine.
- This routine can be slow on large networks.
- Much of this routine can be automated.



```
#!/bin/bash
```



Overview

- Penetration testing often begins with a simple routine.
- This routine can be slow on large networks.
- Much of this routine can be automated.
- So we wrote a tool to help with the automation.

```
      dM.      ~MMMMMMMb.  MMMMMMMMMMM
      ,MMb     MM      Mb /   MM      \
      d'YM.    MM      MM      MM
      ,P`Mb    MM      MM      MM 6MMMMb
      d'  YM.  MM      ,M9      MM MM'  `Mb
      ,P`Mb    MMMMMMM9'      MM      ,MM
      d'  YM.  MM              MM      ,MM'
      ,MMMMMMb MM              MM      ,M'
      d'  YM.  MM              MM      ,M'
      _dM_    _dMM_MM_      _MM_MMMMMMMM
```


Typical Pentest Routine

- Run Nmap (or port scanner of choice)



```
state      service
tcp        open      ssh

to exact OS matches for host

nmap run completed -- 1 IP address (1 host up) scanned
sshnuke 10.2.2.2 -rootpw="210N0101"
connecting to 10.2.2.2:ssh ... successful.
attempting to exploit SSHv1 CRC32 ... successful.
setting root password to "210N0101".
listen open: Access Level <9>
ssh 10.2.2.2 -l root
t@10.2.2.2's password: 
```

The image shows a terminal window with a dark background. The text is green and white. It displays the output of an Nmap scan, showing a single IP address (10.2.2.2) scanned. Below the scan results, it shows the execution of the 'sshnuke' tool with the root password '210N0101'. The tool successfully connects to the SSH service on 10.2.2.2 and attempts to exploit a CRC32 vulnerability in SSHv1, which is successful. It then sets the root password to '210N0101' and opens a listen. Finally, it shows the command 'ssh 10.2.2.2 -l root' and the prompt 't@10.2.2.2's password: '.

Typical Pentest Routine

- Run Nmap *(or port scanner of choice)*
- Review ports and services
 - Port 21 -> test for anonymous FTP
 - Port 80 -> identify web service and check for flaws/default creds
 - Port 445 -> enum users/shares, ms08-067?, ...
 - ...



Typical Pentest Routine

- Run Nmap *(or port scanner of choice)*
- Review ports and services
 - Port 21 -> test for anonymous FTP
 - Port 80 -> identify web service and check for flaws/default creds
 - Port 445 -> enum users/shares, ms08-067?, ...
 - ...
- Run Responder / Metasploit / CrackMapExec / ...



Typical Pentest Routine

- Run Nmap *(or port scanner of choice)*
- Review ports and services
 - Port 21 -> test for anonymous FTP
 - Port 80 -> identify web service and check for flaws/default creds
 - Port 445 -> enum users/shares, ms08-067?, ...
 - ...
- Run Responder / Metasploit / CrackMapExec / ...
- ...
- Take over the DC/database/etc..



If it is not broken...

- Repeatability
- Consistency
- Can be tedious and slow
- Manually parsing through data can be prone to error
- Automation can help



Why Not Use <insert favorite scanner>?

PROS

- Plenty to choose from
- Useful in specific scenarios
- Some are OpenSource / cheap

Why Not Use <insert favorite scanner>?

PROS

- Plenty to choose from
- Useful in specific scenarios
- Some are OpenSource

CONS

- Can be fairly resource intensive
- Can be expensive
- How easy to add a new check/tool?

Automation via Scripting

- Kali already has LOTS of popular tools and scripts
- Automation methods:
 - Bash
 - Python (or scripting language of choice)
 - Metasploit RPC

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase
- Run Nmap and parse output

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase
- Run Nmap and parse output
- Create events based on ports/services

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase
- Run Nmap and parse output
- Create events based on ports/services
- **Modules respond to events to perform specific tasks**

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase
- Run Nmap and parse output
- Create events based on ports/services
- Modules respond to events to perform specific tasks
- **Modules can create new events**

APT2 - Automate the standard stuff

- APT2 is a framework
 - Modules
 - Event queue
 - KnowledgeBase
- Run Nmap and parse output
- Create events based on ports/services
- Modules respond to events to perform specific tasks
- Modules can create new events
- **Runs until event queue is empty**

How Does This Help?

- Multi-threaded event queue is **fast**.
- **Simple to create** new modules for nearly any tool/script.
- Ready to go:
 - Get Kali (or your favorite distro & tools)
 - Clone the repo
- Doesn't have a module for your favorite tool? **Then make it.**

So, What Can It Do?

- Identify services & operating systems
 - Screenshot web applications, X11, VNC, ...
 - Analyze FTP and file shares
 - Brute force accounts
 - Run Metasploit modules
 - Compile hashes -> John the Ripper/HashCat
-
- “ls /usr/share” - If it is listed here, a module can probably be made for it

Anatomy of a Module

- Inherit from base module (typically ActionModule)
- Has standard properties:
 - Name
 - Description
 - Requirements - Which tools need to be installed?
 - Trigger - Which event does this module listen for?
 - Safety Level - Scale of 1 - 5 (5 = safe, 1 = dangerous)
- "process()" is the primary method

Are There Limitations?

- Tools need to be non-interactive

Are There Limitations?

- Tools need to be non-interactive
- Multi-threading is tricky
 - lot of traffic fast
 - modules can have limits defined, depend on the author

Are There Limitations?

- Tools need to be non-interactive
- Multi-threading is tricky
 - lot of traffic fast
 - modules can have limits defined, depend on the author
- Brute force with caution
 - you might break some things
 - safety levels are your friend



Are There Limitations?

- Tools need to be non-interactive
- Multi-threading is tricky
 - lot of traffic fast
 - modules can have limits defined, depend on the author
- Brute force with caution
 - you might break some things
 - safety levels are your friend
- Nonstandard ports and service names may throw off modules

Demo Time

Development

- Open source - Available on the Rapid7 Github account at <https://www.github.com/MooseDojo/apt2>
- Future plans
 - Import from more than just NMAP
 - Responder -> John the Ripper -> Hydra
 - Multiple Passes
 - Pretty Reports
 - ?

411 & Questions

- Adam Compton
 - @tatanus
 - adam_compton@rapid7.com
 - adam.compton@gmail.com
- <https://www.github.com/MooseDojo/apt2>
- QUESTIONS???