Building Cyber Ranges: An Introduction

Donavan Cheah

A Cybersecurity Enthusiast

August 20, 2019

Outline

1. Who am I?

2. Cyber Range Introduction

3. Building a Machine

4. From One Machine to a Range

5. Concluding Remarks

Who Am I?

- uid $\neq$ 0,
- An OSCP holder.
- Cybersecurity chose me: I came from Physics.
- Blog on Digital World musings: `https://donavan.sg/blog`.
- Probably still *trying harder* to be a better cyber professional.
- Vulnerable machines: 5 built so far; search them on Vulnhub.
- Some Interests: penetration testing, musing about the digital world, travel, and still a big STEM enthusiast.

DISCLAIMER!

All material here represents **ONLY my own personal views**. These views do **NOT** represent the view of my current employer, DSTA, neither should they be misconstrued as representation of any other entity except myself.

Why Build Cyber Ranges?

We want to learn in a fun way! (especially the builders).

- A different learning experience as compared to those solving challenges (think CTFs). In particular, very open-ended learning!
- Building (and setting) problems require a different set of skills as compared to breaking them.
- Building multiple different machines and “replicating” an enterprise can provide a platform for learning.

What Can We Build in a Cyber Range?

In theory, for the purposes of training, we can build “anything”.

Examples

- Penetration testing (PT) ranges (also known as the “OSCP-like” range).
- Exploit development ranges (test exploits and telemetry, more research-oriented)
- Blue team training ranges (e.g. IR/forensics, mock-up SOCs e.t.c.)

Disclaimer

Scope will be on PT ranges. Note that ranges can serve multiple objectives, and we will *very briefly* discuss other objectives.

How to Think of Building the Range?

Framing the problem in the context of the PT range:

- **What** skills do we want to train in budding penetration testers?
- **How** do we want to structure the range to build up their skills (Technique-based? Realistic environment? Tools training?)
- **Why** do we want a cyber range? (Safe learning environment? Configurable to organisation's PT needs? Testing new capability, for firms that develop their own tools?)

Useful Reference (start from single machine first):
`https://www.abatchy.com/2019/02/tips-on-creating-vulnerable-vms`

A Word on CTFs

CTFs are built for a short time range. (Typically anything from **8** to **48** hours.)

- CTF problems are excellent for some techniques, but not necessarily for other important elements of a good pentester (e.g. methodology, exploit development, source code review, enumerative skills)
- The problems required to test such skills are different.

The OSCP-Style Range

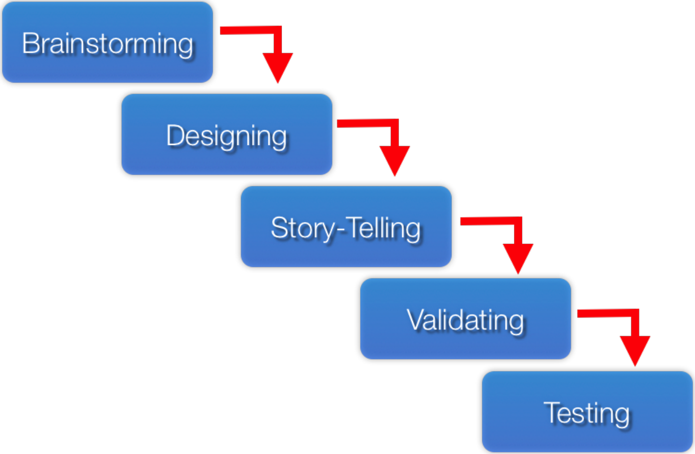
The “OSCP-style range” is an excellent benchmark for penetration testing training (my personal opinion). However, two issues arise.

- What is an “OSCP-style” range?
- What should we cover, and not cover?

Scope

Today, I will only talk about building realistic machines you can find on Vulnhub, HackTheBox e.t.c.. Will not cover other “non-red” team elements such as building blue-team training ranges.

Methodology


Brainstorming

Designing

Story-Telling

Validating

Testing

Framing to build vulnerable machines.

Methodology

Importance of a Different Methodology

Unlike in a penetration test, we are designing "software". We could do it "agile", and iterate, but for purposes of uploading to a site like Vulnhub, let us do this in one shot! Hence a more "traditional" approach.

Methodology

Alert!

Fictitious examples will be used to show the process to avoid spoilers on my Vulnhub machines! For a more detailed account, please scan the QR code or click HERE for full blog post on a machine I built. (Slides to be uploaded after the talk, so you can click on the link.)

Brainstorming

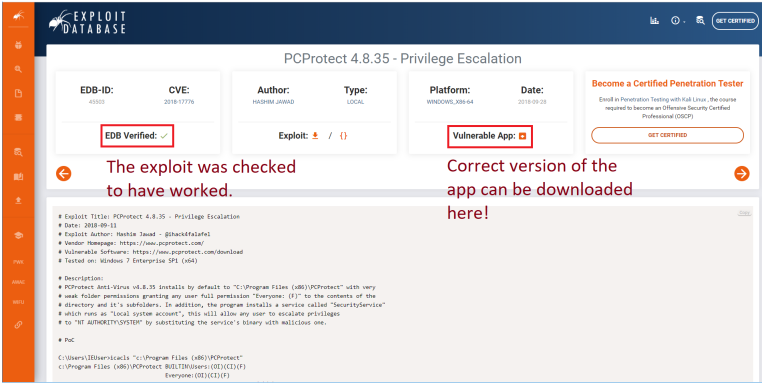
Many types of vulnerabilities and misconfigs to add into machines.

How to Come up with Ideas

Good places to search:

- Exploit-DB
- Other vulnerable machines on VulnHub or HackTheBox
- Look through penetration testing guides. E.g. for privilege escalation: Absolomb's Guide or gotmi1k's Guide

Brainstorming — A Look at Exploit-DB



PCProtect 4.8.35 - Privilege Escalation

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
45503	2018-17776	HASHIM JAWAD	LOCAL	WINDOWS_X86-64	2018-09-28

EDB Verified: ✓

Exploit: ⭳ / {}

Vulnerable App: ⭳

Become a Certified Penetration Tester

Enroll in Penetration Testing with Kali Linux , the course required to become an Offensive Security Certified Professional (OSCP)

GET CERTIFIED

The exploit was checked to have worked.

Correct version of the app can be downloaded here!

```
# Exploit Title: PCProtect 4.8.35 - Privilege Escalation
# Date: 2018-09-11
# Exploit Author: Hashim Jawad - @ihack4falafel
# Vendor Homepage: https://www.pcprotect.com/
# Vulnerable Software: https://www.pcprotect.com/download
# Tested on: Windows 7 Enterprise SP1 (x64)

# Description:
# PCProtect Anti-Virus v4.8.35 installs by default to "C:\Program Files (x86)\PCProtect" with very
# weak folder permissions granting any user full permission "Everyone: (F)" to the contents of the
# directory and it's subfolders. In addition, the program installs a service called "SecurityService"
# which runs as "Local system account", this will allow any user to escalate privileges
# to "NT AUTHORITY\SYSTEM" by substituting the service's binary with malicious one.

# PoC

C:\Users\IEUser>icacls "c:\Program Files (x86)\PCProtect"
c:\Program Files (x86)\PCProtect BUILTIN\Users:(OI)(CI)(F)
                                 Everyone:(OI)(CI)(F)
```

Why is Exploit-DB useful? Allows us to check for known vulnerable programs. Bonus: this is an anti-virus product. Who said security products can't have inherent misconfigurations and vulnerabilities? Link: https://www.exploit-db.com/exploits/45503

Brainstorming — A Look at Exploit-DB

What is Exploit-DB good for?

- Third-party software exploits. (Good source to insert vulnerabilities such as unquoted service path, weak permissions.)
- Checking on the range of vulnerabilities of the products we use. (E.g. if we pick a certain version of a LAMP stack, do we have to watch out for unintended vulnerabilities in the technology stack?)

Brainstorming — Putting the Ideas Together

Typically, in a boot2root machine, we want:

- A method to obtain an initial foothold.
- A privilege escalation method.

Some machines are meant to train technique (e.g. Metasploitable 3 or Lin.Security). These follow different design principles from what we will discuss today, but they are excellent machines for beginners.

Designing

Design Thinking?!

Think realistic. Design as if we have real users. Design realistically. Design is both a science and an art.

- The “science”: How do the exploits work? Putting them together.
- The “art”: How to make the machines look realistic? How to mimic certain types of corporate environment?

Designing – the Science

There must be a flow from boot to root (We are not building machines with no intended solution; we are building intentionally vulnerable machines!!)

Penetration Testing Process

Put one in the shoes of a penetration tester. What steps are required for the machine to be compromised? Write out a proper attack flow.

Designing – the Science

An Example: Simple boot2root flow

1. Start a web server where the CEO, Jane, hints her love for giant cats like tigers, cheetahs and jaguars.
2. Trigger a “forget password” request with a “secret question” such as, “What animal does the CEO enjoy the most?”
3. Fuzz the question and trigger a password reset, gaining access to the web application’s administration portal.
4. Obtain a SSH private key, and log in as jane through SSH.
5. Break out of the jail shell, and realise that Jane can use `vim` with sudo privileges and no password.
6. Use `vim` to spawn a root shell.

Designing – the Art

A realistic-looking environment is important. We want to simulate what real (lazy) users might do. We can add some “rabbit holes”.

Rabbit Holes: Some Ideas

- Set up legitimate non-vulnerable services. Maybe the employee practises bad security architecture and decides to house plenty of services on one server.
- Mimic what real users do. They will interact with applications. Create realistic-looking “junk”.
- You may include various defences. Unlike a CTF, time sensitivity is not as critical.

Story-Telling

It is good to have a theme to the machine to “up” its realism factor.

Examples of Stories

- A corporate IT environment of an SME (Could spin up: print server, web-facing e-commerce page, client machines, network monitoring server, file share server, domain controller)?
- Decide on the role of the machine you build. Especially relevant vis a vis a range once one wants to set up sensible workgroups/domains/VLANs.

Validating

Prepare the machine in a state that is what the penetration tester would be subject to.

What we Must Check For:

- The solution as a boot2root solution. (You may have multiple solutions.)
- The solution is reliable enough. (Some exploits will crash services; we may need to script auto-start for it upon crashing if we want to avoid repeated machine reverts.)
- The mitigation measures implemented on the machine do not successfully stop the boot2root solution.

Validating

We may also have to check abuse cases.

Examples of Abuse

- Poorer error handling than expected.
- Unintended solutions (common with outdated kernels/software, or with introduction of rabbit holes that are unwittingly also vulnerable to some exploit, unintentionally)

Validating

More examples of validation, and weeding out easy abuse methods.

Think from a Red-Teamer's Perspective!

- Make sure to flush out critical logs and history. (e.g. `.bash_history`)
- Look through renmants of development. (Check config files to see that "convenient development artefacts" are no longer there.)

Testing

Probably the **most difficult** part of machine-building.

Why is Testing Difficult

We need to test for:

- Functionality: Any unintended holes?
- User interaction/experience (yes, UI/UX): How do fellow penetration testers interact with the machine?
- Blind spots: Can someone else, not yourself, root the machine?

Testing isn’t That Easy...

We Often Assume Too Much!!!

What usually happens in testing:

- What we thought was difficult isn’t difficult at all for some penetration testers.
- What we thought was easy was devilishly difficult for some penetration testers.

Different penetration testers are skilled in different areas. Our assumptions on what is “easy” or “difficult” can be incorrect.

Testing is Time-Consuming

Faults and Bad Design

Sometimes the issues are more functional in nature.

- Unexpected bugs happen. Different versions of software, custom software that may not always work together without problems.
- We may think our supposedly "good design" is interpreted differently.

Lesson: Penetration testers may not make great UI/UX judgement. Neither are we the best software integrators.

Testing can be Demoralising

REDO!

Sometimes the issues are so catastrophic we'll have to redo some steps.

- Exploit reliability or bugs in exploitation. (Happened to one of my boxes, forcing a rework.)
- We may have created a machine that did not meet our expectations. (Testers do not share your opinion, and the testers share valid opinion.)

Lesson: What we think is great may not be so outside. Just like selling any product.

Testing can be Fulfilling Too

Examples of Fulfilment

Walking through the machines with fellow testers can spark joy.

- Different penetration testers have different thought processes — new tools or approaches at problem-solving.
- Collaborative effort to help improve the penetration testing community!

From One Machine to a Range

With multiple machines in a range, we can introduce realistic network features.

Network Simulation

Some examples of favourites, from a red-teaming angle:

- Client-side exploits (e.g. script a simulated client that visits a page vulnerable to XSS – weaponise an XSS beyond a mere “alert(1)” to target clients in the mock enterprise.)
- Dependencies such as passing-the-hash, training network pivoting through segregating different VLANs

A More Fulfilling Range

Red-teamers are not the only one who find a range useful. Others can partake in the range's activities.

Implementation of Capabilities

What other cyber and non-cyber teams can be involved in:

- Blue team: Table-top exercises for forensics/incident response teams, and threat hunting (leave behind the aftermath of a red team exercise, and contain the problem)
- Education for system administrators, software developers, architects and more: realistic illustrations of indicators of compromise.

More on "blue" versus "red" exercises:
https://securitytrails.com/blog/cybersecurity-red-blue-team

Other Comments on Cyber Ranges

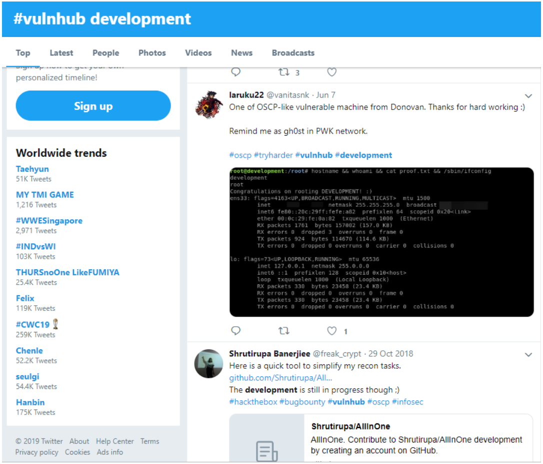
Not One-Size-Fits-All!

The more goals we want to fulfil, the harder it is to fit everyone's goals. Take some care in planning its design, just like a traditional system.

Iterations Between Red and Blue Team

Red teams are useful to inform the blue teams of the weaknesses in defence. Blue teams will then harden their defences in response, forcing the red team to introduce new ways to bypass the more reinforced system.

The Results of the Joy of Building Machines



Sign up now to get your own personalized timeline!

Worldwide trends

Taehyun	51K Tweets
MY TMI GAME	1,216 Tweets
#WWESingapore	2,971 Tweets
#INDvsWI	103K Tweets
THURSnoOne LikeFUMIYA	25.4K Tweets
Felix	119K Tweets
#CWC19	259K Tweets
Chenle	52.2K Tweets
seulgi	54.4K Tweets
Hanbin	175K Tweets

© 2019 Twitter About Help Center Terms Privacy policy Cookies Ads info

laruku22 @vanitasnk · Jun 7

One of OSCP-like vulnerable machine from Donovan. Thanks for hard working :)

Remind me as gh0st in PWK network.

#oscp #tryharder **#vulnhub #development**

```
root@development:/root# hostname && whoami && cat proof.txt && /sbin/ifconfig
development
root
Congratulations on rooting DEVELOPMENT! :)
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet [redacted]  netmask 255.255.255.0  broadcast [redacted]
        inet6 fe80::20c:29ff:fefe:a82  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:fe:0a:82  txqueuelen 1000  (Ethernet)
        RX packets 1761  bytes 157002 (157.0 KB)
        RX errors 0  dropped 3  overruns 0  frame 0
        TX packets 924  bytes 114670 (114.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 330  bytes 23458 (23.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 330  bytes 23458 (23.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

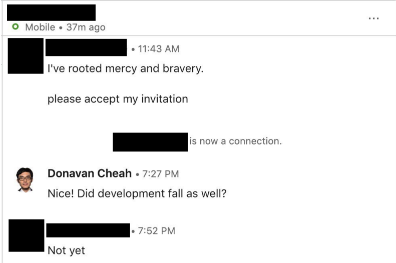
Shrutirupa Banerjiee @freak_crypt · 29 Oct 2018

Here is a quick tool to simplify my recon tasks.
github.com/Shrutirupa/All...
The **development** is still in progress though ;)
#hackthebox #bugbounty **#vulnhub** #oscp #infosec

Shrutirupa/AllInOne
AllInOne. Contribute to Shrutirupa/AllInOne development by creating an account on GitHub.

Phew! At least there was some affirmation that what I declared as "OSCP-like" is really the case.

The Results of the Joy of Building Machines



[redacted]

○ Mobile • 37m ago

...

[redacted] [redacted] • 11:43 AM

I've rooted mercy and bravery.

please accept my invitation

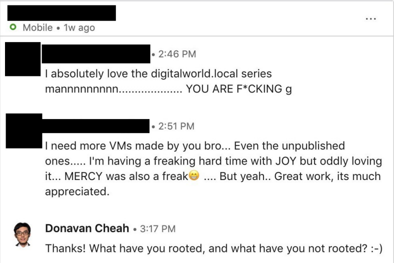
[redacted] is now a connection.

Donavan Cheah • 7:27 PM

Nice! Did development fall as well?

[redacted] [redacted] • 7:52 PM

Not yet



[redacted]

○ Mobile • 1w ago

...

[redacted] [redacted] • 2:46 PM

I absolutely love the digitalworld.local series mannnnnnnnn................... YOU ARE F*CKING g

[redacted] [redacted] • 2:51 PM

I need more VMs made by you bro... Even the unpublished ones..... I'm having a freaking hard time with JOY but oddly loving it... MERCY was also a freak😁 .... But yeah.. Great work, its much appreciated.

Donavan Cheah • 3:17 PM

Thanks! What have you rooted, and what have you not rooted? :-)

I think it is fulfilling when other infosec professionals and budding learners give you a pat on the back!

Cyber Range Fun

Reflections

Building vulnerable machines is a great learning experience.

- Red-teamers: test new PT techniques.
- Blue-teamers: understand configurations, hardening, potential telemetry sources!

Would There be More Machines?

Is five enough? Maybe more? Fewer? Who knows?

The End!

Congratulations for staying throughout!

Contact Details

- Blog: https://donavan.sg is the main landing page for Linkedin, my tech blog and more. Contact me through there.

Slides and a rough transcript of this talk will be uploaded. Find them at https://donavan.sg/materials/div0talk_cyberrange_slides and https://donavan.sg/materials/div0talk_cyberrange_transcript.