Breached Labs

OSCP+ Study Guide.

2025. A Breached Labs Resource.

What is the OSCP+

The OSCP+, an official certification from **Offensive Security**, serves as a prestigious, industry-recognized credential tailored for entry-level **penetration testers** aiming to showcase their practical offensive security skills. This hands-on certification goes beyond theoretical knowledge, demanding a blend of technical expertise and a methodical approach to tackle complex challenges.

Rooted in real-world penetration testing scenarios and methodologies, the OSCP+ equips its takers with the ability to navigate and exploit systems in a controlled, ethical manner.

Recently updated from its predecessor, the classic OSCP, the OSCP+ introduces additional content and heightened requirements, ensuring that certified individuals are well-versed in the latest tools, techniques, and evolving cybersecurity landscape, making it a gold standard for those entering the field.

Ideal Path to the OSCP+

The ideal path to earning the OSCP+ begins with building a strong foundation in the fundamentals, including networking, Linux, and Windows administration, which provide the essential technical groundwork for penetration testing.

From there, those aspiring to pass from the getgo should progress to mastering basic security concepts and familiarizing themselves with commonly used tools in the field. Hands-on practice is critical, and platforms like **TryHackMe** and **HackTheBox** offer accessible vulnerable machines to hone skills in a practical setting.

We would recommend those considering the OSCP to go through the **CRTP** (Altered Security), and **CPTS** (HackTheBox) to have a strong understanding of everything that the OSCP will demand. This will make it a much easier transition into the high level of complexity which you will be introduced to, with the **PEN-200** course.

Enrolling in Offensive Security's PEN-200 course marks a pivotal step, delivering structured learning tailored to the certification's demands. Success requires dedicating **focused** lab time to refine techniques, experimenting with different approaches to exploitation and system compromise.

Equally important is the habit of methodically documenting findings. It's all about developing a personal methodology that ensures clarity and efficiency during testing. Finally, candidates should schedule the exam only when they feel confident in their grasp of the course material and their ability to navigate the lab environments. Never rush, always prepare!

Breached Labs

Career Paths with the OSCP+

Earning the OSCP+ opens doors to a variety of rewarding career paths in the cybersecurity domain, particularly for those with a passion for offensive security. It serves as a gateway to entry-level penetration tester positions, where individuals can apply their hands-on skills to identify and exploit vulnerabilities in systems and networks.

Beyond that, the certification qualifies professionals for security consultant roles, advising organizations on strengthening their defenses, or red team analyst opportunities, where they simulate real-world attacks to test an enterprise's resilience.

For those interested in defensive operations, the OSCP+ also supports roles like Security Operations Center (SOC) analyst, blending offensive insights with monitoring and response duties. It acts as a stepping stone to specialized fields such as web application security (appsec) or cloud security, broadening career horizons.

Alternatively, it can lead to security engineering roles with an offensive security focus, designing robust systems informed by a deep understanding of how they can be breached.

What's New in the OSCP+ Edition

The OSCP+ edition introduces a range of significant updates that enhance its relevance and rigor, aligning it more closely with the demands of modern cybersecurity. A key addition is the inclusion of **Active Directory** attack path requirements, reflecting the critical role that AD plays in enterprise environments and equipping candidates to navigate and exploit these complex systems. This was not the case with the previous PEN-200 curriculum.

The course content has been expanded to focus on modern enterprise environments, ensuring that learners are exposed to contemporary infrastructures and attack surfaces. Updated lab environments now feature current vulnerabilities and scenarios, providing a realistic and up-to-date testing ground for sharpening skills.

The exam format has shifted to a point-based scoring system, offering a more structured and objective evaluation of a candidate’s performance across various tasks. There’s an increased emphasis on privilege escalation techniques, a foundational skill for penetration testers, deepening the focus on moving laterally and vertically within systems.

Additional hands-on challenges have been incorporated, mirroring real-world scenarios to better prepare candidates for practical engagements. Finally, more comprehensive reporting requirements have been introduced, emphasizing the importance of clear, detailed documentation: a critical aspect of professional penetration testing that ensures findings are actionable and well-communicated, much of which will be discussed in this guide.

The Community

The OSCP+ journey is enriched by a vibrant and supportive community that offers a wealth of resources and connections for aspiring penetration testers. The **OffSec Discord** server stands out as a dynamic hub for student collaboration, where learners can exchange ideas, seek advice, and engage with others in real time.[1]

Complementing this, the **OffSec OSCP Forum** provides a dedicated space for in-depth discussions and troubleshooting specific challenges encountered during preparation. You will receive your login credentials alongside a link to the forum whenever you commence any Offensive Security course. These credentials will remain all throughout your journey.

On Reddit, communities like **r/OSCP** and **r/netsecstudents** foster peer-to-peer support, with members sharing study tips, success stories, and practical insights.

LinkedIn groups open doors for networking with certified experts, facilitating mentorship and career opportunities, while GitHub repositories provide

1 https://discord.gg/gUns8B22B7

access to shared tools, scripts, and techniques that enhance hands-on practice. There are plenty of repositories out there with more than enough resources that you can use to pass the OSCP+ exam.

A key ethos within this community is the importance of ethical information sharing. It's all about focusing on guidance and methodologies rather than direct exam spoilers, as to preserve the integrity of the certification process.

In either of these communities, you can request guidance with any issues which may arise, but **do not** be prepared to be spoon fed the solution, but rather directed towards the right path.

The Course (PEN-200)

The **PEN-200** course is the backbone of OSCP+ preparation, crafted to teach penetration testing from scratch to mastery. It covers essential topics like enumeration, exploitation, privilege escalation, and reporting, giving you a full toolkit for offensive security.

The course spans roughly 500 pages across more than 20 chapters, complemented by video lessons that guide you through every step. You'll start with foundational concepts, so think networking and scripting before progressing to advanced techniques like cracking systems and documenting your findings.

Each section wraps up with hands-on labs, where you'll apply what you've learned in simulated real-world scenarios, bridging the gap between theory and practice.

Completing PEN-200 takes about three to six months, depending on your schedule. If you're juggling it with work or life, expect to spend **10-15 hours** a week, though dedicated learners can push through faster. Time management is key; successful students often break the material into daily chunks, like an hour or two, and carve out plenty of time for the labs.

The course starts at a moderate level, approachable for beginners with some tech background, but the complexity ramps up as you dig into tougher exploits and time-pressed challenges.

The labs can be a grind, so do expect late nights troubleshooting and moments of frustration. Students who skip them often regret it, as the hands-on practice is what ties the "why" of the lessons to the "how" of real pentesting.

Overcoming these hurdles builds not just knowledge but confidence, making the difference in the OSCP exam. To get the most out of PEN-200, pace yourself and take good notes, jot down key commands, workflows, and lab successes creates a lifeline for later. Use **Obsidian** for all of your notes-taking. Document every challenge, lab, section. Omit nothing.

The Lab

The OSCP+ lab environment is a **hands-on**, virtual playground built to sharpen your penetration testing skills by throwing you into realistic scenarios. It's designed to feel like a real-world network, with a mix of standalone machines and complex, interconnected systems that challenge you to think like an attacker.

Typically, you'll face around 50-60 lab machines, each varying in difficulty. Whilst some are beginner-friendly to get you started, others are advanced setups that test your creativity and grit. To access the labs, you connect via a **VPN** provided by Offensive Security, giving you a stable link to the environment for the standard 90-day period, or more depending on which access plan you are on (Learn One, Learn Unlimited etc).

If you need more time, resetting your lab access is an option, though it often comes with a fee and restarts your progress, so plan wisely. Only commence the lab and course then when you know you have the time to allocate attention towards it

Working through the labs involves attacking machines, finding vulnerabilities, and submitting "**flags**". These flags are proof of your successful exploits to track your progress. These flags are unique codes hidden on each system, and submitting them confirms you've cracked the challenge, mimicking the proof-of-compromise reports pentesters deliver in the real world.

The labs teach you to build attack paths, pivoting between systems, escalating privileges, and exploiting weaknesses, much like you'd do in an enterprise breach. You'll need to document everything from commands, steps, to findings, not just to stay organized but to prepare for the exam's strict reporting requirements.

There's no set number of labs you *have* to complete, but doing as many as possible is **crucial**. Expect to tackle 30-40 machines in those 90 days if you're consistent, with time management being key. Ideally start with easier targets to build momentum, then carve out focused sessions for the tough ones. Skipping the labs isn't an option if you want to succeed; they're where you turn theory into instinct, refining your approach to enumeration, exploitation, and post-exploitation.

The Exam

The exam is a rigorous, hands-on assessment that tests your penetration testing prowess under real-world conditions, structured around a point-based system designed to evaluate both skill and strategy.

The current exam structure allocates points across various challenges, with a total of **100** points possible: standalone machines each offer up to 20 points (10 for low-privilege access and 10 for full compromise), while the Active Directory (AD) set, comprising of a domain controller and client machines actually carries a hefty 40 points, awarded only upon full domain compromise with no partial credit whatsoever.

Candidates have exactly **23 hours and 45 minutes** to complete the exam, a tight window that demands sharp time management, prioritizing quick wins on standalone targets, allocating roughly 2-3 hours per machine, and reserving 4-6 hours for the intricate AD set, all while factoring in breaks and troubleshooting.

To pass, a minimum of **70** points is required, pushing testers to balance efficiency with thoroughness across the two challenge types: standalone machines, which test isolated exploitation skills, and the AD set, which requires navigating interconnected systems, pivoting, and privilege escalation within a domain environment.

The AD component is rather unforgiving. Complete compromise of the entire set is mandatory for points, emphasizing mastery of enterprise-level attacks like credential harvesting and lateral movement. Documentation during the exam is critical, requiring detailed notes on every step, command, and output, as these form the backbone of the professional penetration test report due within 24 hours post-exam.

This report-writing period, a separate **24-hour** sprint following the hacking phase, demands clarity and precision, with screenshots and replicable instructions to validate each exploit. Unfortunately, failure to meet these strict documentation standards can slash points, making preparation and organization as vital as technical skill.

Mindset

The mindset required is a blend of resilience, discipline, and strategic thinking, epitomized by Offensive Security's "**Try Harder**" philosophy, though applied judiciously to fuel effort rather than frustration. Developing persistence without succumbing to burnout is key; this means pushing through tough challenges while recognizing when to step back and recharge, preserving mental stamina for the long haul.

Learning from failures transforms setbacks into progress. Each unsuccessful attempt reveals a new angle or tool, building a deeper understanding over time. It may take several attempts to pass the exam, but all that really matters is that you pass. Plenty of people have had to sit this gruesome exam countless times, yet never quite gave up.

To avoid spinning wheels, time-boxing stuck periods with a 30-45 minute rule helps: if a solution eludes you, shift focus to another task or seek a hint, then return refreshed rather than grinding endlessly.

Maintaining detailed notes and refining personal methodologies during practice fosters efficiency and clarity, turning chaotic experimentation into repeatable processes. This is a habit that pays dividends in the exam's high-pressure environment.

Confidence grows organically through progressive challenges, starting with simpler machines and scaling to complex networks, reinforcing belief in one's abilities step by step.

On exam day, stress management techniques, like deep breathing, short breaks every 2-3 hours, and a pre-planned schedule keep your nerves in check, ensuring focus remains sharp across the nearly 24-hour marathon.

How to Get Ahead Before Starting

Getting a head start before tackling the OSCP+ involves proactive preparation that builds both technical skills and confidence, setting the stage for a smoother journey through the course and exam. Building a home lab with vulnerable machines, using virtual setups like **VirtualBox** or **VMware** with intentionally insecure systems offers a sandbox to experiment with attacks and refine techniques at your own pace.

Practicing with **TryHackMe's** OSCP-specific learning paths provides structured, beginner-friendly challenges that mirror certification objectives, while completing **HackTheBox's** Starting Point machines introduces hands-on exploitation in a gamified environment, bridging theory to practice.

Studying all public OSCP review blogs from past candidates delivers valuable insights into common pitfalls, effective strategies, and lab experiences, offering a roadmap of what to expect. Learn from everyone else's experience.

Developing efficient documentation methods ahead of time, structured to capture commands, outputs, and screenshots streamlines the reporting process, a critical exam component, and ingrains a professional habit early on. Become familiar with the OSCP reporting standard early.[2]

[2] https://www.offsec.com/pwk-online/OSCP-Exam-Report.docx

Mastering essential tools like **Nmap** for scanning, **Metasploit** for exploitation, and others (e.g., Burp Suite, John the Ripper) ensures familiarity with the penetration tester's toolkit, reducing learning curves during PEN-200.

Finally, creating custom scripts in Python, Bash, or PowerShell for basic repetitive tasks like port scanning or credential testing boosts efficiency and personalizes your workflow, giving you a practical edge before day one of the official journey, so you don't have to dedicate any sacred time in your exam to anything but probing and exploiting.

How to Predict Initial Foothold

Developing a systematic approach to enumeration and exploitation is a crucial aspect of being able to conquer the exam. Crafting systematic enumeration templates, be it actual lists or in your mind: structured checklists or scripts that outline steps like port scanning, service identification, and banner grabbing, ensures that no stone is left unturned, providing a consistent framework for tackling any system.

Recognizing service version vulnerabilities becomes second nature with practice; by cross-referencing version numbers (e.g., from Nmap's -sV flag) against public databases like **CVE** or **Exploit-DB**, testers can quickly pinpoint exploitable weaknesses tied to outdated or buggy software.

Familiarity with common web application entry points, such as login pages, file uploads, or search forms, guides focused attacks, while understanding how to identify vulnerable parameters (e.g., via URL queries, POST data, or cookies) involves testing for injection flaws like SQL, XSS, or command execution using tools like Burp Suite or manual fuzzing. Whilst the OSCP+ is

not so appsec-heavy, it is heavily recommended to become accustomed with the OWASP Top 10.[3]

Checking for default credentials methodically, so starting with vendor-specific lists (e.g., admin:admin, root:toor) across services like SSH, FTP, or web panels, exploits human oversight with minimal effort. **Password spraying** techniques, where a single common password (e.g., “Password123”) is tested across multiple accounts, require careful consideration to avoid lockouts, leveraging tools like Hydra or custom scripts with rate-limiting awareness.

Finally, identifying service **misconfigurations** efficiently, such as overly permissive permissions, exposed debug pages, or unpatched settings, relies on combining tool outputs (e.g., Nmap scripts, Metasploit modules) with a keen eye for anomalies, turning subtle oversights into powerful entry points.

Methodology

Crafting a robust methodology is essential for OSCP+ candidates, enabling them to approach penetration testing with consistency, efficiency, and adaptability across diverse scenarios. Developing a personal penetration testing framework begins with defining a repeatable process tailored to individual strengths, typically encompassing:

- Reconnaissance
- Vulnerability assessment
- Exploitation
- Post-exploitation

[3] https://owasp.org/www-project-top-ten/

Reconnaissance and information gathering procedures kick off the process, involving active and passive techniques like DNS enumeration (using tools like dig), and network mapping to build a comprehensive picture of the attack surface.

Vulnerability scanning with tools like Nmap, followed by manual confirmation steps, such as verifying version-specific flaws or testing for misconfigurations; ensures accuracy beyond automated results, reducing false positives and focusing efforts.

Exploitation strategies vary by system type: standalone boxes might lean on public exploits or buffer overflows, while Active Directory environments demand chaining Kerberos attacks or pass-the-hash techniques, adapting tactics to the target's architecture.

Post-exploitation actions and lateral movement involve harvesting credentials (e.g., Mimikatz on Windows, `/etc/shadow` on Linux), pivoting through compromised hosts, and mapping internal networks to expand control. The privilege escalation workflow splits by OS. On Linux, checking sudo rights, kernel exploits, or writable scripts, while on Windows, targeting misconfigured services, token impersonation, or UAC bypasses, each following a checklist honed through practice.

Network Scans + Active Directory Tricks

Mastering network scans and Active Directory (AD) tricks is a pivotal skill set for OSCP+ candidates, blending precise reconnaissance with sophisticated domain exploitation to unlock complex environments.

Effective **Nmap** scan strategies form the foundation, leveraging timing options (e.g., -T4 for speed without overwhelming targets), scripts (like `--script`

`vuln` or `smb-enum-shares`), and custom options (-p- for full port scans or -sV for version detection) to map networks and pinpoint vulnerable services efficiently.

In AD contexts, **Bloodhound** becomes a game-changer, visually mapping the environment by analyzing trust relationships, user privileges, and attack paths, ingested via tools like SharpHound, as to reveal the shortest routes to domain dominance.

Kerberoasting detection and exploitation involve requesting service principal name (SPN) tickets (using tools like GetUserSPNs.py) and cracking them offline with Hashcat to harvest privileged account credentials, exploiting weak encryption.

Service Enumeration + Attacks

The OSCP+ exam zeros in on service enumeration and attacks, pushing you to turn basic service discovery into full-on breaches by targeting common protocols with precision.

It's all about spotting vulnerable services like SMB, HTTP, SQL databases, NFS, FTP, and IMAP/POP3, and then hitting them with the right tools and techniques. The exam tests your ability to think like an attacker, blending systematic probing with creative exploitation to crack systems, often under time pressure. The most common attack vectors you'll face revolve around misconfigurations, outdated software, and weak authentication, so knowing how to exploit them is key to passing.

For HTTP, the OSCP exam loves web-based attacks. You'll start by **enumerating directories** and files with tools like wfuzz, nikto, dirb, gobuster,

dirsearch (e.g., `dirsearch -u http://target`) to uncover hidden pages or juicy endpoints. From there, SQL injection (SQLi) is a big player. Think of bypassing login forms by injecting payloads like `' OR 1=1 --` into input fields. HTTP enumeration with dirsearch followed by SQLi to dodge auth or pull data is a bread-and-butter combo you'll see a lot. Practice targeting web apps, as they're a staple.

SMB is another OSCP favorite. Enumeration begins with tools like `smbclient -L //target` to list shares or SMBmap to check permissions, exposing weak spots. Attacks then lean on vectors like **EternalBlue** (Metasploit's ms17_010_eternalblue) for unpatched systems, PsExec for lateral movement with stolen creds, or SMB relaying with Responder to grab NTLM hashes.

For **FTP**, test **anonymous** login (ftp target, user: anonymous) and poke configs for flaws, think command injection via crafted inputs or fuzzing with wfuzz might crack it open. IMAP/POP3 enumeration uses Nmap scripts (e.g., imap-capabilities) to map auth methods, with attacks focusing on brute-forcing weak creds via Hydra.

Active Directory Attack Paths

Common attack vectors in the OSCP+ AD labs revolve around privilege escalation and lateral movement. Once you've got a foothold, let's say, a regular user account, you'll have to target high-value accounts or systems by abusing misconfigured permissions or stolen credentials.

Built-in groups like Domain Admins, Enterprise Admins, and Administrators are your bullseye, as cracking them often hands you the keys to the kingdom. A typical move is harvesting credentials with tools like Mimikatz (e.g.,

`sekurlsa::logonpasswords`) from a compromised machine, then using them to hop to a domain controller via **PsExec** or **WMI**. Kerberoasting is another big one, grabbing service account hashes with something like GetNPUsers.py and cracking them offline to escalate privileges.

To nail these paths, you'll start by enumerating the AD environment, so think `net view /domain` or PowerView's[4] `Get-DomainComputer` to list systems, then digging deeper with BloodHound to visualize attack routes.

From there, it's about chaining exploits: pivot from a user account to a server, dump creds, and target group memberships or trusts. The OSCP+ exam loves scenarios where you exploit outdated systems or sloppy configs, like unrestricted admin rights or weak passwords so practice spotting and hitting those.

Enumeration Techniques

The exam tests your ability to probe AD environments systematically, uncovering users, groups, systems, and misconfigurations that lead to big wins. **PowerView** is a go-to tool here, you'll use it to pull detailed lists of domain users (e.g., Get-DomainUser), computers, groups, and trusts, painting a full picture of the network. LDAP queries with tools like ldapsearch (e.g., `ldapsearch -x -H ldap://target`) are another key trick, letting you extract AD objects and attributes fast, whether scripted or manual.

Even old-school net commands on Windows machines, like `net user /domain` or `net group "Domain Admins" /domain` still shine for grabbing user details and group memberships straight from the command line, and will do you well for enumerating on the fly.

[4] https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1

The OSCP+ loves throwing common vectors at you, and **Service Principal Names** (SPNs) are a big one. You'll hunt them with PowerView (`Get-DomainUser -SPN`) to spot service accounts ripe for Kerberoasting. grabbing their hashes and cracking them offline with Hashcat to snag creds.

Then there's the hunt for privileged users. The exam often hides attack paths in accounts with special rights, like Constrained or Unconstrained Delegation. Find them with PowerView (`Get-DomainUser -TrustedToAuth`) and you've got a shot at token impersonation or relaying attacks. To target these, start fairly broad with PowerView or net to map the domain, then zero in on SPNs (Service Principal Names), shares, and delegation flaws.

Credential Harvesting

Snatch sensitive auth data from compromised Windows systems in an AD environment to climb the privilege ladder. One of the top methods is dumping creds from **LSASS** memory via Mimikatz (`sekurlsa::logonpasswords`). Pulling plaintext passwords, NTLM hashes, or Kerberos tickets if you've got admin rights. The exam loves this vector since it's a quick win on poorly secured machines.

Another go-to is raiding the Windows Registry, so target **HKLM\SECURITY** or **HKLM\SYSTEM** with tools like secretsdump.py to grab cached hashes or keys, though you'll need SYSTEM-level access to make it work.

The **SAM** database is a classic OSCP+ target too. Extract local account hashes from `%SystemRoot%\system32\config\SAM` using samdump2 or Mimikatz (`lsadump::sam`), perfect for offline cracking with Hashcat or passing the hash to pivot.

Kerberos tickets are another biggie. Use **Mimikatz** (`kerberos::list` or `sekurlsa::tickets`) to harvest TGTs or service tickets, setting you up for

pass-the-ticket attacks to hop between systems. Finally, don't sleep on Windows Credential Manager. Loot it with Mimikatz (`dpapi::cred`) to snag stored logins from users or apps. To target these, start with a compromised box, escalate to admin or SYSTEM (think MS17-010 or PsExec), then hit LSASS, SAM, or tickets.

Lateral Movement

It's all about leveraging stolen credentials or access to spread your reach, and the exam tests your ability to do it fast and smart.

Pass-the-Hash (PtH) is a go-to vector: grab an NTLM hash with Mimikatz (`sekurlsa::logonpasswords`), then use it with psexec.py or Metasploit's exploit/windows/smb/psexec to authenticate to another box without needing the plaintext password.

Pass-the-Ticket (PtT) ups the game. Snag a Kerberos TGT or service ticket (`kerberos::list`), export it, and inject it with mimikatz' `kerberos::ptt` to impersonate users and slide into restricted systems.

Enumeration Tricks

Tools like **LinPEAS** (Linux) and **WinPEAS** (Windows) are exam MVPs. Run them (./linpeas.sh or winpeas.exe) right after a foothold to scan for

misconfigs, escalation paths, or juicy data like cached credentials. Their color-coded output flags hot spots fast, so focus on red or yellow lines for weak perms or keys.

Custom **wordlists** are another clutch move build them with target-specific terms (scrape key terms from webapps, user accounts etc.) using tools like cewl, then hammer password spraying or bruteforcing (`hydra -L users.txt -P custom.txt`) when generic lists flop (such as **rockyou.txt** from seclists[5]).

SMTP user enumeration is a sneaky OSCP+ favorite. Probe mail servers with VRFY username or EXPN via `telnet smtp.target 25` to confirm valid accounts, feeding AD attacks or phishing prep. DNS zone transfers pop up too, so try `dig @ns.target domain AXFR` or `nslookup -type=AXFR` to snag internal hostnames and IPs if the server's rather poorly configured.

Web enumeration is huge on the exam. Brute-force directories with `dirb http://target wordlist.txt` or `gobuster dir -u http://target -w common.txt` to find hidden pages, tweaking wordlists for tech stacks (add .php, .bak).

Go deeper with content discovery, fuzz params with **ffuf**, hunt backups like index.php.old, or spider with **Burp Suite** to uncover buried endpoints.

Reverse Shell Tricks - MSFVenom

You will be tested on your ability to craft and deliver payloads that lock in access across AD and standalone systems. Setting up versatile multi-handler listeners is step one.

[5] https://github.com/danielmiessler/SecLists

Fire up Metasploit's **multi/handler** (`use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp`) or a **Netcat** combo (`nc -nlvp 443`) to catch reverse shells on multiple ports or protocols. This flexibility is key when you're unsure what'll stick.

Choosing between **staged** and **non-staged** payloads is a big exam call. Staged ones like windows/meterpreter/reverse_tcp split into stealthy chunks for tight networks, while non-staged windows/shell_reverse_tcp dump it all at once, simpler but chunkier.

Custom formats are exam gold. Craft an .exe (`msfvenom -p windows/shell_reverse_tcp -f exe > shell.exe`), .aspx for IIS, or .jsp for Java servers to match the target's stack.

Embedding payloads in files like PDFs or Word docs (`-f pdf --payload windows/meterpreter/reverse_tcp`) exploits file exec flaws or social engineering, just add the -k flag to keep the file functional and sneaky.

Privilege Escalation Tricks

On Windows, **SeImpersonate** token abuse is a star player. Tools like **PrintSpoofer** or **JuicyPotato** (`PrintSpoofer.exe -i -c cmd.exe`) exploit services running as SYSTEM (like Print Spooler) by stealing their tokens, handing you top-tier privileges if SeImpersonatePrivileges is enabled.

Unquoted service paths are another exam gem. Simply spot a service like `C:\Program Files\App\service.exe` with `sc qc servicename`, then drop a rogue `C:\Program.exe` to run as SYSTEM when it restarts, exploiting the missing quotes.

Vulnerable services are fair game too, so enumerate with sc query or WinPEAS, then hit outdated ones (e.g., ms11_080 via Metasploit) to pop a SYSTEM shell.

On Linux, **SUID binaries** are a classic vector, hunt them with `find / -perm -4000 2>/dev/null`, target stuff like passwd or custom apps, and abuse them with **GTFOBins** tricks (e.g., `/usr/bin/passwd -- sh`) for a root shell. Sudo misconfigs are gold: run `sudo -l` to spot commands executable as root without a password, then escalate via `sudo vim` (:!sh) or wildcard flaws (e.g., `sudo tar *`).

Cron jobs are ripe for hijacking. Check `/etc/crontab` or `/var/spool/cron` for scripts or binaries with weak perms you can overwrite, triggering root execution on the next run.
To target these, start with enumeration: WinPEAS/LinPEAS for a quick sweep, then zero in on SUIDs, services, or cron with manual checks.

GTFOBins, Kernel Exploits

GTFOBins leverage built-in Linux binaries like vi, find, or tar to execute discrete commands or escalate privileges without dropping custom tools, making them gold in restricted environments where you're stuck with what's on the box.

Take find as a GTFOBins star. Run `find / -exec /bin/sh \;` on a misconfigured SUID binary, and you've got a root shell if it's set to run as a higher user, slipping past basic monitoring since it's essentially just a "normal" tool.

The trick with GTFOBins[6] is knowing which binaries can be abused. For example, the command less can spawn a shell with !sh, and tar can extract a crafted archive to overwrite files, all using legit system utilities to dodge suspicion.

On the kernel side, Linux exploits like **Dirty Cow** (CVE-2016-5195) are staples. Exploit a race condition to overwrite `/etc/passwd` (dirtycow: /tmp/exploit.c) and grab root, though modern patches might block it, so version matters.

Finding kernel exploit targets starts with safe enumeration. Run `uname -r` on Linux or `systeminfo` on Windows, and then cross-check CVEs on exploit-db.com to avoid crashing the system with untested code. The exam loves throwing vulnerable OSes at you (think Ubuntu 16.04 or Windows 7), so test in VMs first, pre-compile Dirty Cow or tweak a Windows PoC, nail the execution, and save precious time in the 24-hour window.

Stability's critical after kernel exploits, for example Linux might reboot or Windows might BSoD (blue screen of death), so pivot to a stable shell (like Meterpreter migrate) once you've got control.

To target these, start with basic reconnaissance. Run `ls -l /usr/bin` or `find / -perm -4000` to spot GTFOBins candidates, then pair with `uname -r` for kernel plays.

Report Writing

The exam demands detailed, step-by-step documentation, including:

- Every command (whoami, net user)

[6] https://gtfobins.github.io/

- Screenshot
- Output
- Hostname
- Host Address
- Means of patching the exploit
- PoC/Proof Of Concept (if applicable)

Start with a sharp executive summary, keep it short (1-2 paragraphs), skip the jargon, and hit the big picture: scope (e.g., AD pentest), top findings (like domain admin access), and why it matters to the business, all for the hypothetical C-suite's eyes.

Lay out exploitation paths like a story. Initial foothold (SMB relay etc.), privilege escalation (Dirty Cow), lateral movement (Pass-the-Hash), and AD takeover with tools, techniques, and proof, leaving absolutely no gaps.
Screenshots are a must and a frequent OSCP+ trip-up. Snap evidence like a `whoami /all` showing SYSTEM or a domain admin shell, keep text readable, and annotate with arrows or notes (e.g., "Flag: OSCP123") using PNG/JPG format. Structure findings by **risk**: critical, high, medium, low, as such so the report flows logically.

Remediation is where you shine. Don't just say "patch it"; give fixes like "Turn on SMB signing" or "Remove SUID from `/usr/bin/find`" that hit the root issue, showing you understand defense too.

Templates are your friend. We mentioned it before, but use Offensive Security's OSCP one or tweak a Markdown/Word version (quite a few already are available online)[7], pre-filling sections (methodology, findings) to blitz the 24-hour deadline. To nail this, log everything as you go: commands, outputs, screenshots.

[7] https://github.com/noraj/OSCP-Exam-Report-Template-Markdown

Important: Always submit your flags in the control panel, take a picture of hostname + ip + contents of flag.txt. You will miss points by not doing this. Double-check, triple-check!

Training Playgrounds

Picking OSCP-like systems is key to practice, prior to the exam is key. Lean on lists like TJ Null's Hack The Box (HTB)[8] lineup or Offensive Security's **Proving Grounds** (PG) Practice boxes. Retired machines with walkthroughs are a goldmine. Read up on HTB's older retired boxes with IppSec videos[9]. We recommend hitting them blind first, and only then study the solutions to lock in tricks like enumeration or escalation, all without ruining the real exam's surprises.

Time pressure's a big OSCP+ vibe, so set 2-4 hour caps per machine, as to mimic the 24-hour exam grind with a timer, log your start and stop, and practice juggling targets fast to build grit. Variety builds your playbook: you must hit Windows, Linux, web, and AD boxes from HTB or VulnHub to tweak your approach for each, crafting a go-to method for scanning, cracking, and climbing privileges that works anywhere.

For the AD-heavy OSCP+ curve, Pro Labs are your endgame. Whilst not necessarily, you could dive into Offensive Security's PG Play or HTB's **RastaLabs** after a month or two of HTB, spending 1-2 months wrestling with lateral moves, pivots, and domain takeovers in setups that feel like the real deal.

[8] NetSecFocus Trophy Room
[9] https://ippsec.rocks/

It's strongly recommended to become well skilled with both easy and medium boxes prior to engaging in the OSCP+ exam. You'll save yourself a lot of hassle, and increase your pass chances.

Reminders

Breaks are non-negotiable: stick to a rhythm like 50 minutes of grind, 10 off (**Pomodoro** vibes) during practice and the 24-hour gauntlet, stepping back after each machine or big win to dodge burnout and stay sharp.

Notes are your backbone. Get into the habit of logging every move on HTB, VulnHub, or Pro Labs boxes, capturing commands, outputs, and AD attack steps in a tool like Obsidian or CherryTree, so you've got a tight method and report-ready records when it counts.

Cheat sheets[10] [11] will save you a lot of time. Feel free to whip up a doc of your own with go-to lines for nmap, msfvenom, PowerView, or Kerberos plays (think GetNPUsers.py for kerberoasting), searchable and at your fingertips for those clutch moments.

Ending Notes

Ditch memorizing scripts for a method-first mindset. This means approaching each box like a puzzle (scan, crack, climb, write), adapting on the fly when tools flop or setups twist, a must for the exam's curveballs.

[10] https://book.hacktricks.wiki/en/index.html
[11] https://github.com/0xsyr0/OSCP

Gut instinct comes from grind. Start covering HTB, VulnHub, and Pro Labs boxes over months, spotting stuff like SUID issues not from a cheat sheet, but because you've wrestled them enough to feel it, sharpening you for the 24-hour exam.

Documentation's your ace: scribble every command, output, and timestamp as you go in practice, building a reflex that pays off when the exam demands crisp reports fast. Turn those notes into a personal vault. Stash cheat sheets, scripts, and takeaways in Obsidian, OneNote, or a Git repo, so you've got a growing toolkit for OSCP+ and whatever's next.

Don't stop at the cert! Chase OSCE^3, CRTO, or CAPE; keeping pace with AD's latest holes and tools, because this game never sleeps. Live these notes, it's all about method over memory, and endless learning. The OSCP+ becomes your springboard, not your finish line.

Breached Labs