Cybersecurity with Python

Session Goals

- Understand core cybersecurity concepts and the PTES flow at a practical level.
- Recognize common network services (FTP, SSH, HTTP, SMB, DNS, SMTP), what they do, and typical misconfigurations.
- See where Python fits in modern pentest/recon workflows and which libraries are most useful.
- Build and run small Python tools for reconnaissance and enumeration.

PTES in Practice (Concise)

Phase	What You Do (in this session)
1. Pre-engagement/Scope	Define allowed targets & timing; agree on data handling. (We simulate on an instructor-hosted lab.)
2. Intelligence Gathering (Recon)	Discover assets, subdomains, open ports & services; fingerprint tech stacks.
3. Threat Modeling	Map exposed assets to likely vulns (default creds, old versions, weak auth, IDOR).
4. Vulnerability Analysis	Probe inputs/paths; look for misconfigurations.
5. Exploitation	Trigger a benign PoC to prove impact (retrieve a flag, access a harmless endpoint).
6. Post-Exploitation	Collect basic evidence; avoid persistence.
7. Reporting	Record what happened and recommended fixes; clear, reproducible steps.

Quick Networking Refresher

```
[User]  ->  TCP SYN to target:PORT  ->  [Service speaks a protocol]
Examples:
  80/8080 HTTP(S): requests/responses (web apps, APIs)
  22 SSH: remote shell, SFTP
  21 FTP: file transfer (watch for anonymous login)
  53 DNS: records & zone transfers (UDP/TCP)
  25 SMTP: mail transfer (open relays, VRFY)
  445 SMB: file shares, auth negotiation
```

Core Services & Typical Issues

Service	Why It Exists	Common Issues
HTTP(S)	Web apps & APIs	Default creds, directory listing, IDOR, verbose errors, outdated frameworks
SSH (22)	Remote admin & file transfer	Weak passwords, outdated ciphers, exposed keys, banner leaks
FTP (21)	Legacy file transfer	Anonymous login, cleartext creds, writable dirs
DNS (53)	Name resolution	Zone transfers allowed (AXFR), sensitive subdomains
SMTP (25)	Mail transfer	Open relay, user enumeration via VRFY/RCPT TO
SMB (445)	Windows file shares	Guest access, overly permissive shares, info leaks

Python in the Workflow

General: argparse, logging, pathlib
Web: requests, aiohttp (async), beautifulsoup4, lxml
Network: python-nmap (or subprocess nmap), dnspython, scapy, paramiko
Evidence: csv, json, sqlite3; screenshots if you add selenium (optional)

Code Snippets (Copy & Run)

1) Async HTTP Enumerator — aiohttp

Concurrently fetches many URLs with rate-limiting, recording status code, response size, and the Server header.

Use case: First pass over a target list or discovered paths to see what's live, unusual, and what tech leaks via headers.

```
import argparse, asyncio, csv
from pathlib import Path
import aiohttp
def load_targets(p):
    return [x.strip() for x in Path(p).read_text().splitlines() if x.strip()]
async def fetch(session, url):
    try:
        async with session.get(url, timeout=8) as r:
            text = await r.text()
            server = r.headers.get("Server","")
            return url, r.status, len(text), server
    except Exception as e:
        return url, None, 0, f"ERR:{e}"
async def main(args):
    targets = load_targets(args.targets)
    sem = asyncio.Semaphore(args.rate)
    async with aiohttp.ClientSession(headers={"User-Agent":"PyRecon/1.0"}) as s:
        async def bound(u):
```

```
            async with sem:
                return await fetch(s, u)
        rows = await asyncio.gather(*[bound(u) for u in targets])
    with open(args.out, "w", newline="") as f:
        w = csv.writer(f); w.writerow(["url","status","length","server"]); w.writerows(rows)
if __name__ == "__main__":
    ap = argparse.ArgumentParser(description="HTTP enum")
    ap.add_argument("-t","--targets", required=True)
    ap.add_argument("-o","--out", default="out.csv")
    ap.add_argument("--rate", type=int, default=5)
    asyncio.run(main(ap.parse_args()))
```

2) DNS Subdomain Probe — dnspython

Tries candidate subdomains (wordlist.domain.tld) and prints those that resolve to A records.

Use case: Expand scope by finding staging/dev/admin hosts that often have weaker controls.

```
import argparse, dns.resolver
if __name__ == "__main__":
    ap = argparse.ArgumentParser()
    ap.add_argument("-d","--domain", required=True)
    ap.add_argument("-w","--wordlist", required=True)
    args = ap.parse_args()
    resolver = dns.resolver.Resolver()
    resolver.lifetime = 2.0
    for word in open(args.wordlist):
        sub = f"{word.strip()}.{args.domain}"
        try:
            ans = resolver.resolve(sub, "A")
            ips = ", ".join(sorted({r.to_text() for r in ans}))
            print(f"[+] {sub} -> {ips}")
        except Exception:
            pass
```

3) Nmap Driver — subprocess

Runs an nmap scan and saves XML for later parsing/triage.

Use case: Automate scan → parse → follow-up where open ports feed direct checks.

```
import subprocess, sys
target = sys.argv[1]
cmd = ["nmap","-Pn","-sS","-T4","-p","21,22,53,80,443,445,8080","-oX","-","--open", target]
xml = subprocess.check_output(cmd, text=True)
open("scan.xml","w").write(xml)
print("[*] Saved scan.xml.")
```

4) SSH Banner & Auth Policy Check — socket

Connects to SSH and reads the initial banner line to fingerprint versions/policies.

Use case: Spot outdated SSH servers or policy misconfigs without brute force.

```
import socket, sys
ip, port = sys.argv[1], int(sys.argv[2])
s = socket.socket(); s.settimeout(2.5); s.connect((ip,port))
print(s.recv(128).decode(errors="ignore").strip())
s.close()
```

5) Simple ICMP Ping Sweep — scapy (admin)

Sends ICMP echo requests across a subnet and prints responsive hosts.

Use case: Quick LAN discovery before more specific enumeration.

```
from scapy.all import ICMP, IP, sr1
import ipaddress, sys
network = ipaddress.ip_network(sys.argv[1], strict=False)
for host in network.hosts():
    pkt = IP(dst=str(host))/ICMP()
    resp = sr1(pkt, timeout=0.5, verbose=False)
    if resp:
        print(f"[+] Host up: {host}")
```

Practice Platforms

Hack The Box (HTB): Realistic boxes and labs; great for network+web enumeration depth and modern exploits.
TryHackMe: Guided, beginner-friendly rooms with step-by-step tasks; good for structured learning paths.
VulnHub: Downloadable VMs you run locally; perfect for offline/air■gapped practice and custom classroom challenges.
PentesterLab: Badge-based exercises focusing on specific web vulns with crisp explanations and hands-on challenges.
Web Security Academy (PortSwigger): High-quality interactive labs aligned to OWASP topics (XSS, SQLi, SSRF, etc.).
Codewars: Algorithmic kata in Python; keeps coding sharp between security labs (parsing, data handling, quick scripts).

Threats and Attacks

Malware Families

Virus. Code that attaches to files and spreads when those files run.

Example: An infected Word macro that adds itself to other docs on open.

Defend: Disable/limit macros by policy; AV/EDR signatures/heuristics; least-privilege.

Worm. Self-propagating across networks without user action.

Example: A worm scans LAN for SMB vuln and copies itself to each host.

Defend: Patch management; segmentation; IDS/IPS to throttle scanning; SMB hardening.

Trojan. Malicious code disguised as legit software.

Example: “Free PDF converter” installer drops a backdoor.

Defend: Application allow-listing; reputation checks; user training; AV/EDR quarantine.

Ransomware. Encrypts files and demands payment.

Example: User opens a phishing attachment; file shares get encrypted.

Defend: Offline backups + restore drills; least-privilege shares; email filtering; EDR behavior blocks.

Spyware/Keylogger. Steals info/keystrokes and exfiltrates.

Example: Browser extension siphons cookies.

Defend: Browser hardening; extension allow-list; EDR detections; outbound filtering.

Rootkit. Hides presence by tampering with OS internals.

Example: Kernel driver hides malicious processes.

Defend: Secure boot; signed drivers; EDR kernel checks; reimage if compromised.

Social Engineering & Phishing

Phishing. Deceptive messages to steal creds or deliver malware.

Example: Clone Microsoft 365 login page steals passwords.

Defend: Phishing-resistant MFA (FIDO2); DMARC/SPF/DKIM; awareness training; safe-links.

Spear-phishing/Whaling. Targeted phishing to high-value people.

Example: Fake CEO email asks CFO for urgent wire.

Defend: Finance process controls (call-back); role-based simulations; external sender banners.

Pretexting/Tailgating. Human manipulation to gain access.

Example: “I’m with IT—can you badge me in?”

Defend: Physical security policy; staff training; visitor escort rules.

Network Attacks

Man-in-the-Middle. Intercept/alter traffic between parties.

Example: ARP spoofing on open Wi■Fi proxies victims' HTTP.

Defend: HTTPS + HSTS; WPA2-Enterprise; ARP/DHCP protections; VPN on untrusted nets.

DNS Poisoning/Spoofing. Trick users into resolving to wrong IPs.

Example: Rogue DHCP gives malicious DNS; bank.com → attacker.

Defend: DNSSEC validation; secure DHCP; DoT/DoH; allow-listed resolvers.

Web App Vulnerabilities

SQL Injection. Untrusted input becomes SQL.

Example: `id=1 OR 1=1` dumps rows.

Defend: Parameterized queries/ORMs; least-privileged DB user; WAF rules; input validation.

Cross-Site Scripting (XSS). Injected JS runs in victim's browser.

Example: Comment field renders `alert(1)`.

Defend: Output encoding; CSP; input validation; avoid dangerous sinks.

CSRF. Forces a user's browser to perform an action.

Example: Hidden form changes account email on load.

Defend: SameSite cookies; anti-CSRF tokens; re-auth for sensitive actions.

IDOR. Change identifier to access other objects.

Example: `/note?id=0` reveals admin note.

Defend: Enforce authorization per object; avoid exposing direct IDs.

Directory Traversal. Read files via path tricks.

Example: `/download?f=../../etc/passwd`.

Defend: Normalize paths; allow-list; minimal FS permissions.

Command Injection. User input reaches a shell.

Example: `;cat /etc/shadow` appended to a ping command.

Defend: Avoid shelling out; safe APIs; strict allow-lists if shell unavoidable.

SSRF. Server fetches attacker-chosen URLs.

Example: Image fetch to `http://169.254.169.254/…`.

Defend: Egress allow-list; block link-local/internal IPs; sign/validate resource URLs.

Insecure Deserialization. Untrusted serialized data gets loaded.

Example: Python pickle payload executes code server-side.

Defend: Avoid unsafe formats; use JSON; integrity/signing.

Identity & Credential Attacks

Password Reuse/Stuffing. Using leaked creds on another service.

Example: Leaked VPN creds still work.

Defend: MFA; breach monitoring; password managers; block common passwords.

Brute Force/Spraying. Guessing many passwords slowly across many accounts.

Example: Trying 'Spring2024!' across users.

Defend: Rate limiting; lockouts + alerts; MFA; anomaly detection.

Supply Chain & Dependency Risks

Malicious Packages/Typosquatting. Adversaries publish look■alike packages.

Example: `reqeusts` imitates `requests` on PyPI.

Defend: Pin deps; internal mirrors/allow-lists; signed packages; SCA tooling.