📘 Automation in Cybersecurity with Python – 5 Weeks Curriculum

---

WEEK 1 – Python Foundations for Cybersecurity

Class 1 – Python Setup & Basics for Security

Theory:

- Why Python is popular in Cybersecurity (SOC, IR, Threat Hunting, Automation)
- Setting up Python (installation, venv, IDE)
- Python basics: variables, data types, operators, input/output
- Cybersecurity context: how Python replaces repetitive CLI tasks

Practical:

- Install Python & VS Code / PyCharm
- Create a virtual environment
- Write a script to print system information (OS, username, Python version)
- Simple script: check if a given port number is valid (1–65535)

Class 2 – Python Control Flow & File Handling

Theory:

- Control structures (`if`, `for`, `while`)
- Lists, dictionaries (good for storing IPs, logs, credentials)
- File operations (`open`, `read`, `write`, append)
- Using Python for log file parsing basics

Practical:

- Write a script to read a text file and print each line with line numbers
- Parse a mock log file of login attempts → extract "FAILED LOGIN" entries
- Count frequency of failed attempts per IP and write results to a report file

---

WEEK 2 – Networking & Recon Automation

Class 3 – Python Networking with Sockets

Theory:

- What are IPs and ports (TCP/UDP review)
- The Python `socket` library basics
- Security use case: scanning for open services
- Banner grabbing for fingerprinting services

Practical:

- Write a Python script to resolve a hostname (DNS lookup)
- Build a simple TCP port scanner (scan 10 common ports)
- Grab service banner from an open port and print version info

---

Class 4 – Web Automation with Requests

Theory:

- HTTP basics (GET, POST, headers, status codes)
- Using `requests` to interact with web servers
- Automating security checks for web services (e.g., availability, headers)

Practical:

- Write a script to check if a website is online (200 OK vs 404/500)
- Automate downloading and analyzing HTTP headers (check if `X-Frame-Options` or `Content-Security-Policy` is missing)
- Create a script to check a list of URLs from a file and report their status

---

WEEK 3 – Log Analysis & Threat Detection

Class 5 – Advanced Log Parsing

Theory:

- Types of logs (auth logs, firewall logs, web server logs)
- Using Python string methods and regex for pattern matching
- Security use case: detecting brute-force attempts

Practical:

- Parse a sample `auth.log` for failed SSH logins
- Extract usernames and IPs
- Count repeated failed attempts from the same IP
- Write results into `suspicious_ips.txt`

Class 6 – Threat Intelligence & API Integration

Theory:

- What is threat intelligence in cybersecurity?
- Common APIs: VirusTotal, AbuseIPDB, Shodan
- Parsing JSON responses with Python
- Security use case: automated reputation checks

Practical:

- Write a script that reads suspicious IPs from a file
- Query a mock API (or real API like VirusTotal with free key)
- Print whether each IP is malicious, suspicious, or clean
- Save results in a JSON/CSV report

WEEK 4 – Security Automation in Action

Class 7 – Real-Time Monitoring & Alerts

Theory:

- Continuous monitoring (like `tail -f` for logs)
- Python `logging` module for alerts
- Security use case: triggering alerts for repeated attacks

Practical:

- Write a script to watch a log file in real-time
- If failed login attempts exceed a threshold → trigger alert
- Console alert + log alerting into a separate file
- Extension: send email alert using Python `smtplib`

Class 8 – System & Network Task Automation

Theory:

- Running system commands with `os` & `subprocess`
- Checking system security settings (firewall, users, processes)
- File integrity monitoring with hashes (`hashlib`)

Practical:

- Automate execution of `netstat` and `who` commands, save results to a file
- Write a script to calculate file hashes (MD5/SHA256) for integrity check
- Detect if a critical system file has been altered

WEEK 5 – Remote Security & Final Projects

Class 9 – Automating Remote Administration (SSH)

Theory:

- Basics of SSH in security operations
- Using `paramiko` to automate remote commands
- Collecting data from multiple servers

Practical:

- Connect to a remote server (lab VM) using `paramiko`
- Run security commands (`who`, `last`, `netstat`) remotely
- Save outputs in a central report for analyst review

Class 10 – Final Capstone & Review

Theory:

- Recap of Python automation techniques
- Common pitfalls in automation (false positives, error handling)
- How SOCs use automation (SOAR platforms, custom scripts)

Practical (Capstone):

- Build a **Mini SOC Automation Toolkit** that:
    - Monitors logs in real-time for brute-force attempts
    - Extracts suspicious IPs
    - Queries an API to check reputation
    - Logs results into a report
    - Alerts analyst if IP is malicious

✅ **By the end of this program, students will be able to:**

- Use Python confidently for cybersecurity tasks
- Automate network scans, log analysis, and threat intelligence checks
- Write real-time monitoring and alerting scripts
- Perform automated remote administration tasks
- Build a **security automation toolkit** that mimics real SOC workflows