Course : The AI-enhanced Pentester

Local models, AI system attacks, autonomous agents

Practical course - 3d - 21h00 - Ref. HIA

Price : 2470 CHF E.T.

NEW

Traditional pentesting is undergoing profound change. Defenses are becoming automated (EDR ML, SIEM AI, adaptive WAF), and so are attacks. Cecours trains AI-enhanced pentesters, capable of using uncensored local models to accelerate each phase of the pentest, attacking systems integrating AI (chatbots, RAGs, autonomous agents), and building their own pentest agents.The entire training is based on local models (Ollama, Dolphin-3): zero cloud dependency, mission confidentiality guaranteed, air-gappable. All training is based on local models (Ollama, Dolphin-3): zero cloud dependency, guaranteed mission confidentiality, air-gappable. Direct API, no magic framework.

Teaching objectives

At the end of the training, the participant will be able to:

- Install and configure a local and sovereign offensive AI environment (Ollama, uncensored models)
- Use AI to accelerate recognition, payload generation and Active Directory analysis
- Identify and exploit vulnerabilities in systems incorporating LLMs (OWASP Top 10 LLMs 2025)
- Attacking RAG architectures (extraction, poisoning, indirect injection)
- Building a supervised autonomous pentest agent with ReAct loop
- Audit AI-generated code ("vibe coding") and identify typical vulnerabilities

Intended audience

Pentesters, offensive security consultants, SOC/CERT analysts, security auditors, technical CISOs wishing to understand offensive and defensive AI.

Prerequisites

Experience in penetration testing or offensive security (OSCP level or equivalent recommended). Basic knowledge of Python (file manipulation, REST API, subprocess), Linux, Docker and classic pentest tools (nmap, Burp, BloodHound). No prior knowledge of AI or machine learning is required.

PARTICIPANTS

Pentesters, offensive security consultants, SOC/CERT analysts, security auditors, technical CISOs wishing to understand offensive and defensive AI.

PREREQUISITES

Experience in penetration testing or offensive security (OSCP level or equivalent recommended). Basic knowledge of Python (file manipulation, REST API, subprocess), Linux, Docker and classic pentest tools (nmap, Burp, BloodHound). No prior knowledge of AI or machine learning is required.

TRAINER QUALIFICATIONS

The experts leading the training are specialists in the covered subjects. They have been approved by our instructional teams for both their professional knowledge and their teaching ability, for each course they teach. They have at least five to ten years of experience in their field and hold (or have held) decision-making positions in companies.

ASSESSMENT TERMS

The trainer evaluates each participant's academic progress throughout the training using multiple choice, scenarios, hands-on work and more.
Participants also complete a placement test before and after the course to measure the skills they've developed.

Course schedule

1 Why pentesting as we know it is dead

- Understand the impact of modern AI defenses (EDR ML, NDR, SIEM AI, adaptive WAF) on traditional techniques.
- Analyze attacker/defender asymmetry and the need for AI augmentation.
- Differentiate between censored (ChatGPT) and uncensored (Dolphin-3) models for offensive use.
- Understand the challenges of technical sovereignty: mission confidentiality, air-gap, zero telemetry.

Hands-on work

Same prompt sent to ChatGPT (censored) and local Dolphin-3 (uncensored). Comparison of results for offensive payload generation.

2 Fundamental limitations of LLM

- Understand LLM architecture as applied to pentesting: tokens, context window, temperature, system prompt.
- Integrate Shannon's limit: LLM compresses with loss, it doesn't create information.
- Identify hallucinations and their impact (invented CVEs, approximate orders).
- Know your blind spots: business logic vulnerabilities, creative chains, environmental context, adversarial reasoning.

3 Installation of the offensive AI laboratory

- Install Ollama and the 4 templates (dolphin3, qwen2.5-coder, nomic-embed-text, llama3.2).
- Create a custom "pentester" model with Modelfile and offensive system prompt.
- Deploy Open WebUI for the graphical user interface.
- Validate the environment and test in air-gap mode.

Hands-on work

Complete installation of the offensive AI environment. Creation of custom model. Validation: payload generation and identification of at least one hallucination.

4 AI-enhanced recognition

- Understanding what LLM brings to recognition: synthesis, correlation, prioritization.
- Identify limitations: LLM does not have access to the network, nor does it check for vulnerabilities.
- Recognition agent architecture: subprocess (nmap, curl) + LLM analysis.
- Use the appropriate temperature: 0.3 for analysis, 0.7 for payloads.

Hands-on work

Recognition Python mini-agent. nmap scan of a Docker target, LLM analysis of results, automatic generation of a prioritized attack plan.

TEACHING AIDS AND TECHNICAL RESOURCES

• The main teaching aids and instructional methods used in the training are audiovisual aids, documentation and course material, hands-on application exercises and corrected exercises for practical training courses, case studies and coverage of real cases for training seminars.
• At the end of each course or seminar, ORSYS provides participants with a course evaluation questionnaire that is analysed by our instructional teams.
• A check-in sheet for each half-day of attendance is provided at the end of the training, along with a course completion certificate if the trainee attended the entire session.

TERMS AND DEADLINES

Registration must be completed 24 hours before the start of the training.

ACCESSIBILITY FOR PEOPLE WITH DISABILITIES

Do you need special accessibility accommodations? Contact Mrs. Fosse, Disability Manager, at psh-accueil@orsys.fr to review your request and its feasibility.

5 Payload generation and WAF bypassing

- Compare static payloads (SecLists) vs. generative payloads (LLM).
- Implement an iterative loop: generate ? test ? feedback ? adapt.
- Use feedback from WAF responses (403, blocked patterns) to guide LLM.
- Evaluate the syntactic quality of the payloads generated and the success rate.

Hands-on work

Adaptive fuzzing with LLM. Automated SQLi/XSS payload generation loop against a target with WAF. Bypass rate analysis vs. manual approach.

6 AI and Active Directory

- Understand the problem of large-scale AD analysis (10,000+ objects).
- LLM feeding strategy: Python pre-filtering of SharpHound/BloodHound data.
- Automatic extraction: Domain Admins, Kerberoastable, AS-REP Roastable, dangerous ACLs, delegation.
- Generation of prioritized attack paths with impacket/crackmapexec commands.

Hands-on work

Analysis of LLM-simulated BloodHound data. Python pre-filtering, identification of attack paths, comparison with BloodHound visual analysis.

7 AI-enhanced social engineering

- Understanding AI as a force multiplier for social engineering.
- Generate personalized spearphishing pretexts at scale.
- Identify the obsolescence of traditional anti-phishing training in the face of AI.
- Apply a strict ethical framework: red team only, no deepfakes.

Hands-on work

Simulated red team against MedTech Solutions (scenario provided). Generation of targeted pretexts for 3 profiles, Office 365 phishing page, paired detection exercise.

8 OWASP Top 10 LLM & Agentic AI

- Master the OWASP Top 10 risks for LLM Applications (2025).
- Understanding specific risks OWASP Top 10 for Agentic AI (December 2025).
- Prompt Injection (LLM01): direct and indirect injection, extraction techniques.
- System Prompt Leakage (LLM07): role-play, encoding, multi-turn, crescendo, side-channel.

Hands-on work

Attack on a vulnerable chatbot with 4 levels of increasing protection. Extraction of flags through increasingly sophisticated prompt injection.

9 Attacking RAG systems

- Understand RAG architecture: embedding, vector base, similarity search.
- Identify the 4 attack surfaces: poisoning, extraction, indirect injection, search manipulation.
- Understand why RAG poisoning is particularly dangerous (persistent influence).
- Implement defenses: access control, document sanitization, anomaly detection.

Hands-on work

Exploitation of a vulnerable Docker RAG. Confidential document extraction, poisoning of vector database, indirect injection via HTML comment.

10 Anatomy of an AI agent and its attack surface

- Understanding AI agent architecture: LLM Core + Tools + Memory + Planner.
- Differentiating chatbot (responds) vs agent (acts) and security implications.
- Identify Agentic attack surfaces: Tool Poisoning, Memory Manipulation, Excessive Agency, Goal Manipulation.
- Understand the impact of a prompt injection on an agent (RCE, exfiltration, phishing).

Hands-on work

Exploitation of the FinBot financial agent. Privilege escalation, SQL tool abuse, database exfiltration, indirect injection via email.

11 Building a supervised autonomous pentest agent

- Understanding the ReAct pattern (Reasoning + Acting) for pentest agents.
- Justify the choice of direct API vs. frameworks (LangChain, CrewAI): control, auditability, stability.
- Implement mandatory human-in-the-loop and blacklist of dangerous commands.
- Generate a structured summary for the pentest report.

Hands-on work

Construction and use of a supervised pentest agent. ReAct loop against a Docker target, human validation of each action, automatic final summary.

12 Vibe coding from a safety perspective

- Understand the security risks of AI-generated code (vulnerable patterns in training data).
- Identify the 8 typical vulnerabilities: SQLi, XSS, IDOR, unfiltered upload, hardcoded secrets, path traversal, cleartext passwords, lack of CSRF.
- Integrate "vibe coded" code auditing into a pentest service.
- Compare human auditing with LLM auditing of the same code.

Hands-on work

Pentest of a note management application entirely generated by AI. 8 intentional vulnerabilities, 100 points. Bonus: LLM audit of the same code and comparison.

13 AI Red Team CTF

- Challenge 1 - The chatbot (30 pts): credential extraction via prompt injection.
- Challenge 2 - The poisoned base (40 pts): confidential document extraction + poisoning RAG.
- Challenge 3 - The manipulated agent (50 pts): privilege escalation, tool abuse, DB exfiltration.
- Challenge 4 - The complete chain (60 pts): classic web + prompt injection + agent = RCE.
- Bonus Challenge - The shield (20 pts): reverse challenge, securing a chatbot against automated attacks.

Hands-on work

CTF in teams of 2-3 people. Complete Docker infrastructure with real-time scoreboard. 200 points in total. All tools and techniques from days 1 and 2 are used.

14 Debriefing and outlook

- Position an "AI Red Team" service (sales pitch, deliverables, OWASP guidelines). Integrate the AI audit into a classic pentest methodology (scoping, recon, exploitation, report).
- Understanding the outlook: MCP Security, multi-modal agents, deepfakes, EU AI Act.
- Identify monitoring and training resources (Gandalf, Dreadnode Crucible, PortSwigger LLM Labs, AI Goat).