Slide 1

Hello, welcome everyone to this hands-on course, "Practical Penetration Testing with the Montimage Pentesting Toolbox." My name is Edgardo Montesdeoca from Montimage.

I will be guiding you through this curriculum. It is an intermediate course of less that 20 hours, across five modules with a final hands-on assessment.

This course is part of the CyberSuite project, which is co-funded by the European Union and focuses on implementing innovative Security-as-a-Service solutions to protect modern digital infrastructures.

Slide 2

Agenda Course is as follows:

Our journey is organized into several key phases. We will begin with a general course introduction and then move into Module 1, Pentesting Foundations, Ethics and Methodology.

In Module 2: we will get started with MMT-Pentester tools.

Module 3: will explain reconnaissance and present Attack Scenarios.

While Module 4: will focus on Monitoring, Detection and Reporting.

We conclude with Module 5: presenting Advanced techniques using AI Agents for Tool Integration; before moving to a final capstone assessment and course wrap-up.

Slide 3

Let's start with the course introduction section.

Slide 4

First I will provide some brief comments to set the overall picture.

Penetration testing is how organisations find their weaknesses before attackers do.

In this course you will learn the complete penetration-testing lifecycle — reconnaissance, controlled exploitation, monitoring, and reporting — and how to carry it out safely and ethically using the Montimage MMT-Pentester toolbox, a web platform that brings recon, attack simulation, monitoring, and reporting together in one guided interface.

A note on safety and ethics: Everything is hands-on, but every exercise runs only against provided, authorised lab targets. The platform launches real attack tooling. You must only use it against the lab targets provided for this course, with explicit authorisation.

Unauthorised testing of systems you do not own or have permission to test is illegal.

Slide 5

By the end of this course, you will have a solid understanding of the penetration-testing lifecycle and the legal and ethical rules of authorised testing.

You will be able to perform reconnaissance with Nmap and map a target's attack surface.

Configure and run controlled attack scenarios (for instance, SSH brute-force attacks and HTTP content discovery (dirb) attacks) in a safe lab.

Monitor executions in real time and interpret intrusion-detection alerts. Generate professional reports and write actionable remediation recommendations. Finally you will understand how AI-assisted testing can be used and the learn more on the platform's integrated tool ecosystem.

Slide 6

This course suits IT and cybersecurity students, junior SOC analysts, system and network administrators, and developers who want practical penetration-testing skills.

A basic understanding of networking (for instance, IP addresses, ports, HTTP, SSH) is recommended, and feeling comfortable with command line execution.

No prior penetration-testing experience is required — every concept is introduced before it is used.

Slide 7

The course is organised as five modules plus a final hands-on assessment. Each module is a short lesson followed by a quiz:

You can follow the course at your own pace; the full effort is less than 20 hours, ideally spread over several days.

Slide 8

Recommended procedure to follow is: View each video in order (or go through the slide set and read the text).

For each video, pause it if some term or concept is not clear for you, and search the web or ask a chat bot for more details.

Active participation is important for acquiring higher knowledge.

For Modules 1 to 5 and Capstone, follow the instructions to carry out the exercises or answer the quiz.

At the end, provide your feedback. This is important for us to be able to improve the course.

Slide 9

To earn your digital certificate of completion from the CyberSuite Academy, you must pass each module's quiz with a score of at least 7 out of 10 and complete the final capstone assessment.

Before we begin, a brief note on ethics: when working with network traffic, you may see sensitive information. You must treat all data as confidential and follow your organization's data handling policies. The PCAP files we provide for the labs have been sanitized for educational safety. The tools should be used only for educational purposes or applied only on authorized targets.

Slide 10

I will now present Module 1 in 3 Parts.

Part 1 presents the foundations of Penetration Testing.

Slide 11

First, I will define penetration testing and explain why organisations need it.

A penetration test (or "pentest") is an authorised, simulated attack on a system, network, or application, carried out to find and demonstrate exploitable weaknesses or vulnerabilities so that they can be mitigated before a real attacker exploits them.

It answers a blunt, practical question: if someone tried to break in, could they — and how far would they get?

The goal is constructive, not destructive. A pentest puts in evidence a real risk, proves the impact of that risk, and hands defenders a prioritised list of fixes.

Crucially, it produces evidence rather than opinion: a demonstrated exploit ("we logged in and read the customer table") is far more persuasive to management than a scanner label that simply says "high severity."

Pentesting must be performed under controlled, authorised conditions — it is not random hacking, and not just a list of theoretical issues.

Slide 12

Penetration testing matters for several connected reasons.

First, it lets you find weaknesses before attackers do, discovering exploitable flaws while you still control the outcome.

It also validates your defences, confirming that firewalls, patching, and monitoring actually work under attack rather than just on paper.

Beyond that, it helps you prioritise risk, turning a long list of theoretical issues into a short list of what truly matters and that needs to be rapidly mitigated.

Finally, it supports compliance and trust: frameworks such as PCI-DSS, ISO 27001, and SOC 2 expect regular testing, and clients and regulators increasingly ask for proof that it has been done.

Slide 13

You must distinguish between vulnerability scanning, penetration testing, and red teaming.

Vulnerability scanning is an automated, broad sweep that lists potential weaknesses without exploiting them.

Penetration testing is a human-driven, scoped engagement that confirms weaknesses by safely exploiting them to show real impact.

Red teaming is a goal-oriented, stealthy adversary simulation that tests people, process, and technology together against a realistic attacker.

Beginners often blur three related security activities together. They sit on a spectrum of depth, realism, and cost — not three unrelated jobs. A common mistake is to treat a scan report as if it were a pentest.

The distinction is concrete: a scanner says "port 22 may allow weak authentication"; a pentest proves it by logging in. Red teaming goes further still — it is goal-oriented

adversary emulation that tests an organisation's detection and response across people, process, and technology, often stealthily and over weeks.

Slide 14

You must also be aware that there are different knowledge-depth models (black/grey/white box) that could be involved.

This model describes how much inside information the tester is given before starting. There is no "best" box colour — only the most appropriate one for the engagement's goal. Grey-box is the practical default because it spends time efficiently without pretending the attacker knows nothing.

Pentesters must walk through the phases of an engagement, from pre-engagement scoping to reporting; and apply the legal and ethical rules — authorisation, scope, rules of engagement, and responsible disclosure — that make testing legitimate.

Slide 15

A pentest follows a repeatable lifecycle. It is a cycle, not a strict one-way pipeline: findings in a later phase frequently send you back to an earlier one (new access reveals new things to scan).

Pre-engagement defines the scope and authorisations.

Reconnaissance includes scanning and enumeration

Exploitation involves carrying out the attacks.

Post-Exploitation allow analyising the results obtained and provide feedback and loop on new findings.

Reporting provides all the results and analysis for further investigating remediation measures that are needed, and support compliance and trust.

Pre-engagement always comes first: scoping, rules of engagement, and written authorisation. This is the non-technical foundation that makes everything legitimate.

Reconnaissance gathers information about the target. Passive reconnaissance uses public data with no direct contact; active reconnaissance directly probes the target. Good targeting beats blind exploitation, so this is where engagement time is well spent.

Slide 16

Now I will provide more detail on each phase.

Reconnaissance and Scanning gather information about the target. There is Passive reconnaissance that relies on public data and does not involve any direct contact. On the other hand, Active reconnaissance involves directly probing the target.

Scanning and enumeration map the live attack surface that includes hosts, open ports, services, versions and likely entry points.

Output provides a clear picture of what exists and where the soft spots might be.

In the labs you will use Nmap to enumerate services and map a minimal attack surface against a provided lab target.

Slide 17

The next phase includes Exploitation and Post-Exploitation.

Exploitation safely uses a weakness to gain access or cause a defined effect, proving the risk is real.

For Post-exploitation, once an attacker gains access, you assess what an attacker could actually do, such as, pivot, escalate privileges, access data — all within scope.

Restraint is needed to avoid actual damage (especially in systems under operation). This requires to stay in scope, and document every action.

Lab examples that will be provided are an SSH brute-force against a discovered service, and an HTTP content-discovery scan with dirb against a discovered web service. These techniques are very popular (for the attackers, of course) and easy to exploit.

For instancen dirb, or Directory Buster, is a command-line web content scanner. It works by launching a dictionary-based attack against a target web server and attempting to locate existing directories and files by making HTTP requests based on pre-defined wordlists.

Slide 18

Reporting is the phase that turns activity into value; for the client it is often the most important part of the whole engagement.

A complete report is layered. It opens with an executive summary that states the risk in plain language for decision-makers.

It then gives the technical findings, documenting each issue with evidence, a severity rating, and clear steps to reproduce it.

It closes with prioritised, genuinely actionable remediation guidance, and appendices recording scope, methodology, tools, and raw output.

In MMT-Pentester much of this is produced for you: the platform generates PDF and Excel reports directly from your executions.

In the platform both the PDF and the Excel come from the same report record, so the narrative and the data never disagree.

Slide 19

Before any hands-on work, one principle governs everything: you operate only with explicit authorisation and inside a defined scope — the rules of engagement.

In this course that means testing only the provided, isolated lab targets, and never real or third-party systems.

Stay within scope, minimise harm, protect any data you encounter, and document what you do.

And remember that unauthorised testing is illegal in most jurisdictions, whatever your intent. This is non-negotiable.

If you are ever unsure whether something is in scope, treat it as out of scope until you have written confirmation.

Slide 20

At this point you have the mental model: why we test, the different types of testing, and the phases of an engagement.

Next we introduce the MMT-Pentester Dashboard and its data model — Project, Campaign, and Scenario, with targets and attacks.

You will then run the three guided scenarios hands-on and observe the results on the dashboard.

Keep one idea in mind throughout: every tool and every click maps back to a phase you have just learned.

From here the course becomes hands-on, so each concept you have just met will reappear as a concrete action in the tool.

Slide 21

We now move to Module 1, Part 2: Legal, Ethics and Rules of Engagement.

In this part, the rules that make testing lawful come before any tool.

Slide 22

We cover law and ethics before any tool for a good reason: penetration testing means deliberately attacking a system, and the only thing separating that from a crime is permission.

Put plainly, a skilled tester with no authorisation is a liability, whereas an authorised tester with modest skills is an asset.

Everything in MMT-Pentester therefore runs only against provided, isolated lab targets, with explicit authorisation — never against real or third-party systems.

It is one of the few fields where the very same action is either a paid service or a crime, decided entirely by permission — which is why this foundation matters more than any tool.

Slide 23

Authorisation means explicit, written permission from someone who genuinely owns or controls the target.

A verbal “go ahead”, or an email from the wrong person, is not authorisation — you need it in writing, from someone entitled to sign off.

That permission must exist before the first packet is sent, and it must cover exactly what you intend to do.

The rule is simple: no authorisation, no test.

That permission must name the specific systems, time windows, and techniques you are allowed to use.

Slide 24

Unauthorised access — even “just scanning” — can breach computer-misuse laws such as the CFAA in the United States, the Computer Misuse Act in the United Kingdom, and their equivalents across the EU.

The consequences are real: criminal charges, civil liability, fines, and a damaged career.

“I was only curious” or “I didn’t change anything” is not a defence; what matters legally is intent and access.

Consent and a signed agreement are what turn the same technical act into a legitimate service.

The safeguard is simple and non-technical: never touch a system you cannot show written permission to test.

Slide 25

A proper engagement rests on a written agreement: a contract or Statement of Work, together with an authorisation letter — sometimes called the “get-out-of-jail” letter.

Its key clauses cover who authorises the work, what is permitted, limits on liability, confidentiality, and how findings will be handled.

Keep the authorisation letter to hand throughout the test as your proof of permission.

And remember that consent can be withdrawn: if the client says stop, you stop.

Agree in advance how findings and evidence will be stored, shared, and destroyed — treat it as a professional relationship, not just paperwork.

Slide 26

Scope defines exactly which assets you may touch: IP ranges, hosts, domains, applications, and accounts.

Anything not explicitly listed as in-scope is out-of-scope by default — when in doubt, leave it alone.

The rules of engagement define the how and when: time windows, allowed techniques, prohibited actions, rate limits, and escalation contacts.

Commonly prohibited items include social engineering, physical access, destructive payloads, and anything that touches production data.

For example, a web server may be in scope while the database behind it is not; the boundary is whatever the client wrote down, not what seems technically related.

Slide 27

During a test you will encounter sensitive data — credentials, customer records, internal configurations. Treat all of it as confidential.

Collect only what you need to prove a finding: capture minimal evidence such as screenshots, hashes, or redacted samples, rather than exfiltrating real data.

Store evidence and reports encrypted and access-controlled, and delete them according to the agreed retention period.

The platform follows the same discipline: change default credentials, use a strong session secret, run behind HTTPS, and restrict network access.

The guiding rule is a minimal footprint: prove the finding, record just enough evidence, and leave the data where you found it.

Slide 28

If you discover a genuine vulnerability — in MMT-Pentester or anywhere else — report it responsibly, and do not post it publicly.

For MMT-Pentester the process is to email security@montimage.com, and never to open a public issue.

Include a clear description, steps to reproduce, an assessment of impact and severity, and any mitigations you can suggest.

A typical timeline acknowledges the report within about forty-eight hours, assesses it within a week, fixes critical issues within thirty days, and then discloses in coordination.

Coordinated disclosure protects users, because the fix can land before the details become public.

Slide 29

The platform is intended for educational and research use, in authorised and controlled environments only.

Using it against systems you neither own nor have explicit permission to test is illegal and outside the scope of this project.

The three guided scenarios — Nmap recon, an SSH brute-force, and an HTTP content-discovery scan with dirb — run only against the provided lab target.

Before each exercise, confirm the same three things: is this target authorised, in scope, and within the allowed time window?

Slide 30

We close Module 1 with Part 3: Methodologies and Frameworks.

Slide 31

A methodology is a repeatable, agreed plan for how a test is run — not simply which tools you use.

Its benefits are practical: full coverage so you do not forget steps, repeatability, results you can compare, and a shared vocabulary with clients and defenders.

Without one, testing degenerates into poking at random things until something breaks — unscoped, unauditable, and hard to report.

A phase-based plan is also what you agree with the client, so methodology ties directly back to scope and authorisation.

It is also what makes two testers' results comparable, and what lets a client trust that nothing was skipped.

Slide 32

Four frameworks recur in this field, and they overlap on purpose.

PTES describes what a tester does — the end-to-end engagement process, in seven phases.

The OWASP Testing Guide zooms into web-application specifics and pairs with the OWASP Top 10.

The Cyber Kill Chain and MITRE ATT&CK describe the attacker's side: the Kill Chain as seven linear steps, ATT&CK as a knowledge base of real-world tactics and techniques.

A simple way to hold them together: PTES and OWASP describe what the tester does, while the Kill Chain and ATT&CK describe what the attacker does.

Slide 33

PTES lays out seven phases in order, and the bookends matter: it starts with authorisation and scope, and ends with a report.

The phases run from pre-engagement, through intelligence gathering, threat modelling, and vulnerability analysis, into exploitation and post-exploitation, and finally reporting.

MMT-Pentester's Project–Campaign–Scenario structure and its automatic reports line up with the intelligence, exploitation, and reporting phases.

The lesson is that PTES tells you the order of work and what each phase should produce — the discipline this whole course is teaching.

In practice you rarely march through all seven in a straight line — a later finding often sends you back to gather more information.

Slide 34

The OWASP Testing Guide is a structured catalogue of web tests — information gathering, configuration, authentication, session management, and input validation. Reach for it when the target is a web application.

It pairs naturally with the OWASP Top 10: the guide is how to test, the Top 10 is what commonly goes wrong.

The Cyber Kill Chain models a single intrusion as a line of seven steps — recon, weaponise, delivery, exploit, install, command-and-control, and actions.

It is deliberately defender-friendly: break any one link and the whole attack stops.

For defenders this is empowering: because the steps are sequential, disrupting any single link — detection, patching, or segmentation — defeats the whole intrusion.

Slide 35

ATT&CK separates two ideas. A tactic is the adversary's goal for a step — the "why" — such as Reconnaissance, Initial Access, Credential Access, or Impact.

A technique is how they achieve it — the "how". One tactic holds many techniques, and a technique can serve several tactics.

A sub-technique is a more specific variant of a technique.

Unlike the Kill Chain's fixed seven steps, ATT&CK is granular and non-linear, which is why it maps so well to real engagements.

This is the single most important distinction in ATT&CK, and it is why the same technique can appear under more than one tactic depending on the attacker's goal.

Slide 36

ATT&CK identifiers follow a simple format: the letter T, a four-digit technique number, and optionally a dot with a three-digit sub-technique.

For example, T1595 is Active Scanning under Reconnaissance, and T1499 is Endpoint Denial of Service under Impact.

Reading one further, T1110 is Brute Force, and T1110.001 is its Password Guessing sub-technique.

Because these identifiers are stable, cite them in your reports so clients and defenders can look up detection and mitigation directly.

Citing the ID rather than a vague label lets a defender jump straight to MITRE's documented detections and mitigations for that exact behaviour.

Slide 37

The platform's flow is recon, then attack, then report — which is simply PTES compressed into intelligence gathering, exploitation, and reporting.

Each guided Scenario you run corresponds to one or more ATT&CK techniques exercised against an authorised lab target.

The Reports view, with its PDF and Excel output, is your PTES phase-seven deliverable; tag the findings there with their ATT&CK IDs.

So it is one workflow seen through three lenses: PTES gives the order, ATT&CK names what you did, and the report communicates it.

The three guided scenarios are the spine of the hands-on labs, each standing in for one link of that recon–attack–report chain.

Slide 38

The three labs in this course map cleanly onto ATT&CK.

Scenario one, Nmap recon, is Reconnaissance — Wordlist Scanning, T1595.003.

Scenario two, the SSH brute-force, is Credential Access — Brute Force, specifically Password Guessing, T1110.001.

Scenario three, the dirb HTTP content discovery, is Reconnaissance — Wordlist Scanning, T1595.003. Together they form a mini kill chain: recon to find the door, credential access to open it, content discovery to map it — always against the isolated lab target.

Slide 39

We now begin Module 2: Getting Started with MMT-Pentester.

In this module you will meet the platform, its architecture, and how it organises work.

Slide 40

The MMT-Pentester Dashboard — the Montimage Pentesting Toolbox — is a web platform for planning, orchestrating, and executing security tests and attack simulations.

Instead of running scanners and attack tools by hand in separate terminals, you drive them all from one browser interface and watch the results stream back live.

It is built for controlled labs: every test runs against provided, isolated targets you are authorised to test.

Think of it as mission control for an authorised engagement — you define the work, launch it, watch it, and report on it.

The problem it solves is fragmentation: instead of juggling separate tools, terminals, and notes, you plan, launch, observe, and report from a single place.

Slide 41

You do not need to install anything to follow this lesson; the goal here is the mental model of how the pieces fit together.

The front end is a React single-page app in your browser, and the back end is an Express and Node.js API that handles logins, storage, and starting the tools.

A MongoDB database stores projects, campaigns, scenarios, users, and audit logs, while WebSocket channels push events and raw tool output live, with no need to refresh.

Around this sit the external tools, which run as separate processes, and an AI assistant built on MCP and Together AI that analyses their output.

The front end is a React single-page application, so navigation feels instant and updates arrive without reloading the page; you do not need to memorise the internals, only which part is responsible when something behaves unexpectedly.

Slide 42

Your browser sends ordinary requests to the REST API — “log me in”, “create a project” — which in development lives at localhost, port 3000.

For anything live, two WebSocket channels run in parallel: Socket.io carries campaign and scenario events, and a raw channel on port 9090 carries the tool output itself.

The API starts each external tool as a child process and streams its output back over that channel in near real time.

Alongside the API sits the AI layer — MCP with Together AI — which analyses results and powers the autonomous Agent.

In short, the REST API handles the “do this” requests, while the WebSocket channels handle the “here is what is happening now” stream.

Slide 43

Every fresh install ships with a single super-admin account: username admin, password admin123456.

On first login you are required to change that default password — every install, and the first thing you do.

Behind the scenes the platform hashes passwords with bcrypt, tracks sessions by IP, and locks an account after five failed logins.

Take this seriously: the default credentials are public knowledge, so leaving them in place is the same as having no password at all.

Because those defaults are documented and identical on every install, changing the password is the first real security decision you make on the platform.

Slide 44

MMT-Pentester organises everything behind five views, reachable from the left-hand navigation.

The Dashboard gives a status overview and quick entry points, and Projects is where engagements live, with their campaigns and team access.

Scenarios is where you define and run tests — targets, attacks, and live output — and the Agent view offers the AI-assisted autonomous mode.

Finally, Reports holds the automatically generated PDF and Excel documents. These five views are the map you navigate throughout the course.

As a beginner you will spend most of your early time in Projects and Scenarios — that is where engagements are built and run.

Slide 45

The single most important concept for finding your way around is how work is organised.

A Project is the top-level container for an engagement or client, and a Campaign is a phase or themed batch of work inside it.

A Scenario is one concrete test: it binds targets — each a host, port, and protocol — to attacks with their parameters.

Scenarios can run sequentially or in parallel, and their output streams live over WebSocket.

Concretely, a Project such as “Acme Lab Assessment” holds a Campaign like “External Recon”, which in turn holds a Scenario like “Nmap scan of the lab target”.

Slide 46

MMT-Pentester uses role-based access control, so your permissions depend on your assigned role.

There are three roles, in order of power: superadmin, admin, and user.

A superadmin has full control, including other admins and global settings; an admin manages users and projects within their scope; and a user works inside the projects they are granted.

Every action is written to an audit log, so who did what is always traceable.

Most learners work as a single super-admin on their own install; roles matter most on shared, multi-tester deployments.

Slide 47

New accounts are created through invite-code-gated registration: you need a valid code to sign up.

This prevents strangers from creating accounts on a platform that can launch real attack tooling.

For tighter control, admins can disable registration entirely and create accounts by hand.

On this course your instructor provides the invite code; self-paced learners simply use the default super-admin account.

Slide 48

You now hold the whole mental model: one browser app, backed by an API, a database, live WebSocket feeds, the external tools, and an AI assistant.

You can log in, secure the account, navigate the five views, and understand how work is organised.

The lab for this part is small but foundational: create your first Project and Campaign, the containers everything else lives in.

After that, the following modules fill those containers with the three guided scenarios and teach reporting.

Get comfortable creating these containers now, because from here on every scan and attack you run lives inside a Scenario.

Slide 49

We move into Module 3, Part 1: Reconnaissance and Enumeration.

In this part you gather information about a target and turn it into a map of what is worth testing.

Slide 50

Reconnaissance, or recon, is the first phase of an engagement: gathering information about a target before touching it hard.

The goal is to build an accurate picture of what exists — hosts, services, versions — so that later phases are targeted rather than blind.

Good recon saves time, because you exploit the soft spots you actually found instead of guessing.

Footprinting is the broad early sub-step: sketching the outline — address ranges, exposed services, technologies — before zooming in.

Recon is also what keeps you efficient and in scope: you learn what exists before you decide what to touch.

Slide 51

Recon comes in two flavours. Passive recon gathers information without touching the target — public DNS, WHOIS, search engines, certificate logs, leaked data. It is low-risk and hard to detect.

Active recon probes the target directly — pinging it, scanning ports, grabbing banners. It is faster and more accurate, but it generates traffic the target can log.

Nmap scanning is active recon, so both the target and any monitoring can detect it.

The rule of thumb: start passive to scope the work, and go active only inside an authorised, isolated lab.

Passive sources include WHOIS and DNS records or certificate logs; active means you are sending packets the target itself can see and record.

Slide 52

One reminder before any scanning: every scan in this course runs only against the provided, isolated lab target.

Never scan real or third-party systems — even “just a port scan” can be unlawful, and it is easily detected.

The difference between a pentester and an attacker is permission and scope, not technique.

MMT-Pentester runs tools against the targets you define in a Scenario, so keep those targets firmly inside the lab.

Slide 53

Nmap, the Network Mapper, is the standard open-source tool for active recon and enumeration, and its output is the backbone of your attack-surface map.

It answers three questions, in order.

First, host discovery: is the target up? Then port scanning: what doors exist?

And finally service and version detection: who answers the door? Those answers feed directly into the phases that follow.

Each question narrows the picture: from which hosts are alive, to which ports are open, to exactly what software is answering.

Slide 54

A handful of flags do most of the work. The -sV flag detects the service and version behind a port, while -p- scans all sixty-five thousand ports at the cost of time.

You can also name specific ports with -p, run default safe scripts with -sC, limit yourself to host discovery with -sn, or skip discovery entirely with -Pn.

A common, thorough combination is nmap -sC -sV -p- against the target.

That single command gives you versions and default-script output across every port — a solid enumeration baseline.

When you only need the two services this course uses, a faster nmap -sV -p 22,80 against the target is enough.

Slide 55

Each result line follows the same pattern: port, state, service, and version.

The version column, produced by -sV, is the one that tells you exactly what to look up next.

The state matters too: “open” means a service is accepting connections — those are the interesting ones; “closed” means reachable but nothing is listening; “filtered” means a firewall is blocking and Nmap cannot tell.

Always record the open ports and services verbatim, because that record is your attack surface.

Reading the state correctly matters: “filtered” is not “closed” — a firewall is hiding the port, which is itself useful intelligence.

Slide 56

An attack surface is the set of exposed services an attacker could try to reach.

For each open port, ask three things: what is this service, what version is it, and is it a known entry point?

Prioritise services that commonly allow access — SSH on port 22 invites credential attacks, while HTTP and HTTPS invite web attacks or denial-of-service.

In this course the two services you are looking for are SSH and HTTP; the goal is a minimal attack-surface map — a short, ranked list of services worth testing, with reasons.

In this course two services will stand out — SSH for a credential attack and HTTP for the web attack — so your map should put those first.

Slide 57

In the platform, recon runs as a Scenario under a Project and Campaign, with targets and attack parameters.

The steps are straightforward: create or open a Project, add a Scenario, use the reconnaissance-with-Nmap template, set the lab target, and execute.

As it runs, the live tool output streams to you over WebSocket, so you see Nmap's results as they arrive.

Afterwards you review the results in the Reports view, and the outcomes also appear on the dashboard.

Keep the shorthand in mind: targets are the "where", attack parameters are the "how", and the Scenario is the runnable unit that ties them together.

Slide 58

To recap: recon means gathering information first, and it splits into passive, with no contact, and active, direct probing — Nmap being active.

The Nmap workflow is host discovery, then port scan, then service and version detection.

You read the output by port state and version, and turn the open services into a ranked attack surface.

Your deliverable is a short recon report identifying the SSH and HTTP services for the attack scenarios that follow.

Slide 59

We continue into Module 3, Part 2: Running Attack Scenarios.

In this part you turn your recon findings into two controlled, observable attacks.

Slide 60

In the previous part you ran Nmap recon and mapped a minimal attack surface — a live host offering an SSH service and an HTTP service.

This part turns those findings into action, with two guided attack scenarios: an SSH brute-force and an HTTP content-discovery scan with dirb.

As always, everything runs only against the provided, isolated lab target. Authorised use only.

Slide 61

Before any attack, one rule stands above the rest: run attacks only against the provided, isolated lab target, and only with explicit authorisation.

Never run a brute-force or a denial-of-service against real, production, or third-party systems — it is both illegal and harmful.

A real denial-of-service can take a business offline, and real credential attacks are unauthorised access.

The lab exists precisely so you can learn these techniques safely and legally.

Slide 62

Recall the hierarchy: Project, then Campaign, then Scenario, and within a Scenario, targets and attacks.

A target is a host, port, and protocol — what you are testing.

An attack is a tool invocation with its parameters — how you are testing it.

A Scenario binds one or more targets to one or more attacks, and then executes them.

Keep the mental shorthand: targets are the “where”, and attacks with their parameters are the “how”.

Slide 63

Configuring a target is a matter of three fields.

Set the host to the lab target’s IP or hostname from your recon results, and the port to the discovered service port — 22 for SSH, 80 for HTTP.

Set the protocol to the service type, such as SSH, HTTP, or TCP.

Targets are reusable across attacks within the same Scenario, so you define them once.

Slide 64

To configure an attack, pick the tool and then fill in its parameters — wordlists, thread count, rate, duration.

These parameters control the attack's intensity and behaviour, and sensible defaults are provided for the lab.

Each attack is bound to a target, so the tool knows exactly where to point.

For the labs the provided defaults — such as the supplied username and wordlist — are chosen to work against the lab target, so you can focus on running and observing.

Slide 65

A Scenario can run its attacks in two modes.

Sequential runs them one after another: predictable, easier to read, and gentler on the target.

Parallel runs them at the same time: faster, closer to real multi-vector pressure, but noisier in the output.

While you are learning, choose sequential; switch to parallel when you want to model a realistic combined assault.

Slide 66

The Dashboard streams tool output live over two channels: Socket.io for campaign and scenario events, and a raw WebSocket on port 9090 for the tool's own output.

You watch attempts, responses, and errors scroll past in real time, with no need to wait for a final report.

After the run, the same outcomes reappear as stored results, together with any monitor or IDS alerts.

What you are watching is literally the tool's own output piped to your browser, line by line, as it happens.

Slide 67

The first lab is a brute-force, or credential, attack: repeatedly trying username-and-password pairs until one authenticates.

It targets weak, default, or reused credentials on the SSH service you discovered on port 22.

As it runs you will see authentication attempts stream live, and the detection of repeated failures.

In ATT&CK terms this is Credential Access — Brute Force, T1110, and specifically Password Guessing, T1110.001.

On MMT-Pentester this lab uses Hydra, a fast, widely used credential-guessing tool, pointed at the discovered SSH service.

Slide 68

The second lab is HTTP content discovery. dirb requests many common paths from the web server to enumerate hidden resources, by sending headers very slowly, exhausting the server's connection pool.

It uses very little bandwidth for a large effect: the service degrades or stops responding to legitimate users.

You will watch the open connections climb, response times worsen, and IDS alerts fire.

In ATT&CK terms this is Reconnaissance — Wordlist Scanning, T1595.003.

Because it drips out partial requests, it can exhaust a server with very little traffic — which is exactly why it is worth learning to detect.

Slide 69

Penetration testing is half attack and half observation, so watch the dashboard closely: the traffic seen, expected versus actual behaviour, and any IDS or monitor alerts.

For the brute-force, expect bursts of failed authentications, perhaps lockout or throttling, and an alert on repeated attempts.

For dirb, expect a burst of requests to many paths, discovered resources, and a web-scanning or anomaly alert.

Capture evidence as you go — this is the graded deliverable.

The blue-team view is what turns this from "running a tool" into a learning exercise: you are checking whether a defender would have seen you.

Slide 70

The platform automatically generates reports in two formats: PDF, via Puppeteer, and Excel, via ExcelJS.

Use those alongside your own screenshots to assemble your evidence.

A good run is reproducible: the target, the attack, its parameters, the observed output, and the detection are all recorded.

Slide 71

To recap: you configured targets and attacks, chose an execution mode, and watched the live output.

You ran a Credential Access attack, the SSH brute-force, and a web content-discovery attack with dirb (Active Scanning), and observed how each was detected.

Next comes the deeper analysis and reporting of what these scenarios produced.

Slide 72

We now turn to Module 4: Monitoring, Detection and Reporting.

In this module you learn to read what a test actually did, and to turn it into a professional report.

Slide 73

The earlier modules launched scenarios; this module is about reading the result.

Every execution produces three things worth watching: the live tool output, the dashboard state, and the detection alerts.

Keep the guiding principle in mind: a pentest is only as valuable as the report that comes out of it, and evidence beats opinion.

Launching a tool is the easy part; the value of a pentest lies in the reading and the write-up.

Slide 74

During an execution the Dashboard shows the campaign and scenario status — queued, running, completed, or failed.

Two live channels feed the view: Socket.io for events, and the raw WebSocket on port 9090 for the tool's own output.

Watch the status badges, the live log pane, and the per-target progress as the test runs.

The live panes only update while you hold an active connection, because they are fed by the WebSocket stream.

Slide 75

The single most valuable analytical habit in this course is to write down what you expect to happen before you run something.

Each scenario has an expected signature: Nmap lists ports, the SSH brute-force logs authentication attempts, and dirb (HTTP content discovery) makes the service slow or time out.

“Actual” is what the live output and the target’s response really show.

A mismatch is information in itself: a brute-force with zero authentication attempts means the target was unreachable, not that it was secure.

Predicting first turns you from a spectator into an analyst: you are testing a hypothesis, not just watching output.

Slide 76

An IDS, or Intrusion Detection System, is software that inspects traffic or system behaviour and raises alerts on suspicious patterns.

Detection is mostly of two kinds: signature-based, which matches a known bad pattern such as a burst of SSH logins, and anomaly-based, which flags deviations from a normal baseline.

On the Dashboard these alerts appear alongside the execution output, and they answer a key question: would a defender have noticed your attack?

Think of an IDS as a defender’s smoke detector; you do not configure one here, but you must understand what its alert is telling you.

Slide 77

This is the heart of the module. A finding is not “I think the SSH is weak”; it has five required parts, and the platform’s report model mirrors them.

It needs a title that states the issue in one line, and a severity on the four-tier scale of critical, high, medium, or low.

It needs evidence — screenshots, output, hashes — and a statement of impact: what the issue means for the business.

And it needs remediation: a specific, verifiable fix.

Evidence is the part beginners most often skip — and it is precisely what turns a claim into a finding.

Slide 78

Severity reflects how bad a single finding is, while the report also rolls the findings up into an overall risk score.

A statistics block counts the findings by tier — critical, high, medium, and low.

Judge severity by impact combined with ease of exploitation, not by how impressive the attack felt.

A quiet, easily exploited flaw with real consequences outranks a dramatic one that almost never triggers.

The platform rolls individual severities into an overall risk score and counts findings by tier, giving the client a one-glance sense of exposure.

Slide 79

The platform renders a PDF using Puppeteer, a headless Chrome, producing a polished, client-ready document.

It contains an executive summary, the attack narrative, the methodology, the per-finding detail, a remediation plan, and statistics.

You download it from the Reports view, and its filename follows a clear pattern that includes the target and the date.

It is the narrative deliverable — the document you would actually hand to a client.

Slide 80

The platform also exports structured data as an Excel workbook, with JSON and CSV options.

Where the PDF is the narrative, the Excel file is the machine- and tracking-oriented deliverable: one row per finding, sortable and filterable.

Use it to triage — sort by severity, assign owners, and track remediation status over time.

Both the spreadsheet and the PDF come from the same report record, so they never disagree.

Slide 81

A remediation a busy sysadmin cannot act on without further research is barely a remediation at all.

Compare a weak “harden SSH” with a strong “disable password authentication, enforce key-based login, and add fail2ban with a five-attempt lockout”.

A good entry is specific, verifiable, and prioritised, and the platform stores that priority, the fix, the estimated effort, and the business impact.

Always pair the what with the why, so the owner can justify doing the work.

Prioritise by impact and likelihood, so the owner fixes the most dangerous, most reachable issues first.

Slide 82

A good report is reproducible: someone else should be able to follow your evidence and see the same thing.

And reporting is not the end of the engagement — it is the start of the fix.

The full loop runs from running the attack, to observing live, comparing expected against actual, capturing evidence, writing findings, generating the PDF and Excel reports, and recommending fixes.

Once fixes are applied they are verified by re-testing, and the loop begins again.

Reproducibility is the real test of a report: another tester should reach your conclusion from your evidence alone.

Slide 83

Our final teaching module is Module 5: Advanced — Autonomous Agents and Tool Integration.

In this module you meet the Agent view and the AI layer that assists an orchestrated run.

Slide 84

The earlier modules walked through the three guided scenarios that you drove step by step.

This module looks at the Agent view, where the platform can coordinate a run and use AI to suggest the next step.

The goal is to understand the concept and the tool ecosystem, without overstating what the AI can do on its own.

And the same reminder applies: everything here runs only against provided, isolated lab targets, with explicit authorisation.

Slide 85

The word “autonomous” can sound as though the platform hacks for you. It does not.

An autonomous agent is simply a loop: run a tool, observe the output, analyse it, decide a reasonable next action, and repeat.

In MMT-Pentester the human stays in control — the agent proposes, and you review and authorise before each step.

Think of it as assisted orchestration that reduces busywork between obvious steps, not hands-off hacking.

The loop is run, observe, analyse, recommend, decide — and the “decide” step is always yours.

Slide 86

The Agent view is one of the five platform views you already know, alongside Dashboard, Projects, Scenarios, and Reports.

It is where you start, watch, and review an AI-coordinated session against a lab target.

It uses the same data model as everything else — Project, Campaign, Scenario, with targets and attacks.

And, like the guided scenarios, its live tool output streams to you over WebSocket.

Slide 87

Behind the Agent view sits a backend service called AgentOrchestrator, the coordinator of an autonomous run.

It launches and sequences the tool executions, collects their output, and routes that output to the AI layer for analysis.

It manages the run's state and feeds results back into the platform, so you see progress and recommendations in the Agent view.

It works with the same execution machinery used elsewhere — sequential or parallel runs, and live WebSocket output.

Conceptually it is the coordinator that connects three things: the tools, the AI layer, and the interface you watch.

Slide 88

This module has a clear set of objectives.

You will understand what an autonomous agent is in this platform, and the human-in-the-loop model that keeps you in control.

You will see how AgentOrchestrator coordinates a run, and how MCP with Together AI turns raw tool output into readable insight.

And you will learn the integrated tool ecosystem — what each external tool is for — so you can choose the right one by purpose.

Slide 89

Raw scan and attack output is not, by itself, advice — something has to read it and reason about it.

MCP, the Model Context Protocol, is the standard interface that hands structured context — the tool output — to an AI provider.

An MCP service connects to the MCP server, and Together AI is the provider that performs the analysis.

The effect is that raw output becomes a readable "here is what I see, and here is what you might do next", surfaced back to you through AgentOrchestrator.

The value is translation: raw output becomes a plain-language reading of what was seen and what to consider next.

Slide 90

AI recommendations are grounded in the observed output, but grounded does not mean correct — they can be wrong, generic, or incomplete.

Always validate a suggestion against what you actually see: the live output, the IDS and monitor alerts, and the dashboard observations.

You decide whether to accept, adjust, or ignore each suggested next step.

And recommendations never override the rules: authorised scope and safety always come first.

Treat every recommendation as an input to your judgement, not an instruction to follow.

Slide 91

MMT-Pentester runs a set of external security tools as child processes, and the agent can orchestrate them; each answers a different question.

Caldera handles adversary simulation, while MAIP injects attacks into network traffic.

Shennina automates exploitation, and the KNX Smart Fuzzer and GAN Fuzzer test robustness against unexpected input — one for building automation, one with machine-learning-generated inputs.

And MAG, the MMT-Attacker, generates attack traffic and payloads.

Each tool answers a different question, so choosing the right one matters more than running them all.

Slide 92

Choosing a tool is a matter of matching it to the question.

For adversary simulation and TTP chains, reach for Caldera; for injecting attacks into network traffic, MAIP.

For automated exploitation experiments, Shennina; for protocol robustness and unexpected input, the KNX Smart Fuzzer in building-automation settings, or the GAN Fuzzer for machine-learning-generated inputs.

And for generating attack traffic or payloads, MAG, the MMT-Attacker.

Slide 93

Here is how the pieces combine in a single session.

You pick a lab target and start a session in the Agent view; AgentOrchestrator runs an initial step, such as recon, streams the output, and sends it via MCP to Together AI.

The AI returns an analysis and a suggested next step — perhaps testing a discovered service.

You review the recommendation, decide, and let the loop continue, observing the results on the Dashboard and in Reports.

Slide 94

To recap: the Agent view and AgentOrchestrator coordinate AI-assisted runs, while MCP and Together AI provide the analysis and next-step recommendations.

The tool ecosystem spans simulation, injection, exploitation, fuzzing, and traffic generation — you pick by purpose.

Human-in-the-loop control and authorised scope always apply; the AI's suggestions are inputs, not orders.

Next comes the exploration lab, where you run or describe an agent session and capture its recommendations.

Slide 95

We arrive at the Capstone: the Final Hands-On Assessment — an end-to-end guided pentest.

Here you assemble the whole workflow into one end-to-end engagement.

Slide 96

This is the final module, where you run a complete mini-engagement end to end against a fresh lab target.

The arc is familiar: recon with Nmap, pick two services, run two attacks — the SSH brute-force and the HTTP content discovery (dirb) — observe the detections, write the final report, take the final quiz, and earn your certificate.

Everything you need you have already learned; today you assemble it under your own steam.

The lab is graded against a rubric, so work methodically and document as you go.

Slide 97

You have heard this in every module, but pause on it here, because the capstone is where you work most independently.

The lab target and this exercise are your authorisation and your scope — run everything only against that provided, isolated host, never real or third-party systems.

Minimise harm, document every action, and stop if anything behaves unexpectedly outside the lab.

Unauthorised testing is illegal regardless of intent; the licence to act comes only from explicit permission.

Slide 98

Here is the brief. Your target is a fresh lab host provided for the capstone, with its address given to you.

Your mission is to map the attack surface, demonstrate two distinct weaknesses, and report on the risk with prioritised fixes.

Your constraints: use only the three guided scenario types, observe the results on the dashboard, and produce a single report.

Your output is that final report — the PDF and Excel export plus your written analysis — together with a completed final quiz.

In short, you are a junior pentester given a fresh, authorised host and a short, scoped test to plan and run.

Slide 99

Keep this four-step workflow in view as you work.

First, recon: create a Project, Campaign, and Scenario, and run Nmap. Second, select: read the enumeration and choose the SSH and HTTP services.

Third, attack: run the SSH brute-force, then the HTTP content discovery (dirb).

And fourth, observe and report: read the alerts, then generate and annotate your report.

Slide 100

A rubric scores five areas, where “excellent” means complete, evidence-backed, and clearly communicated.

The five areas are recon, execution, observation, report quality, and recommendations.

The final quiz pass threshold is seventy percent — seven out of ten; pass the quiz and submit the deliverable to complete the course.

Passing then triggers the platform’s certificate flow and your certificate of completion.

The report and recommendations carry as much weight as the technical execution — communication is part of the job.

Slide 101

Now it is over to you. Open the Dashboard at the lab URL, log in, and begin with recon.

Take screenshots of the key moments — scan results, attack output, alerts — as your evidence.

Use the submission checklist in the supporting material before you hand in.

And when the lab is done, take the final quiz to unlock your certificate. Good luck!

Slide 102

That brings us to the course wrap-up and evaluation.

Slide 103

Congratulations on completing Practical Penetration Testing with the Montimage Pentesting Toolbox.

Along the way you have learned the penetration-testing lifecycle and the legal and ethical rules of authorised testing.

You performed reconnaissance with Nmap and mapped a target’s attack surface, then configured and ran controlled attack scenarios.

And you monitored executions in real time, interpreted intrusion-detection alerts, and generated professional reports with actionable remediation.

Slide 104

Please take a few minutes to complete our short course-evaluation survey; your honest feedback helps us improve the lessons, the labs, and the tools for future learners.

The survey asks about the clarity and usefulness of each module, and whether the hands-on labs were easy to follow and worked as expected.

It also asks about the pacing and difficulty, any technical issues you met, and features you would like to see.

The survey is anonymous and takes about five minutes. Thank you for helping us improve — and good luck applying your new skills, responsibly and only on systems you are authorised to test.

Slide 105

Thank you for your attention. This course was brought to you by Montimage.

For any questions, the team is reachable by email, and you can find out more at montimage.eu.