



GHOST
PROTOCOL

LW

THE FIELD LIBRARY

COMPANION

OWN IT, OR BE INVITED

The Lab Workbook

Building your lab, the web labs, network
and AD labs, cloud and container labs, and
the capstone engagement.



Ghost Protocol (Pvt) Ltd · Colombo · ghosts.lk

Security internship training · The Field Library · Edition 1 · 2026

THE GHOST PROTOCOL FIELD LIBRARY · COMPANION

The Lab Workbook

Building your lab, the web labs, network and AD labs, cloud and container labs, and the capstone engagement.

Ghost Protocol (Pvt) Ltd · Colombo

The Field Library, Edition 1 · 2026

Copyright 2026 Ghost Protocol (Private) Limited. All rights reserved.

Sold as a PDF for the buyer's own reading. Please do not share or resell the file.

The library

The Field Library is a whole shelf, not one book. The Core Path is the spine everyone reads. The rest is depth and breadth, taken on as the work calls for it. The shelf is complete now, and you are reading the one marked here.

A · THE CORE PATH

O The Oath BUILT

I Groundwork BUILT

II The Web BUILT

III The Wider Field BUILT

IV The Work BUILT

V The Craft BUILT

B · DEEP TRACKS

AI and Agents BUILT

Reading Code BUILT

Internal and Active Directory BUILT

The Web, Deeper BUILT

Cloud and Containers BUILT

Mobile and APIs BUILT

Reverse Engineering BUILT

Hardware and the Edges BUILT

C · THE OPERATIONS

Red and Purple Team BUILT

OSINT and Recon BUILT

Report Writing Mastery BUILT

The Consulting Craft BUILT

Bug Bounty BUILT

D · THE OTHER SIDE

Detection and Response BUILT

Threat Hunting and Forensics BUILT

Hardening and Architecture BUILT

E · THE GROUND RULES

The Law, Deeper BUILT

Standards and Compliance BUILT

Ethics and Disclosure BUILT

F · COMPANIONS

The Reference Pack BUILT

The Lab Workbook YOU ARE HERE

Case Files BUILT

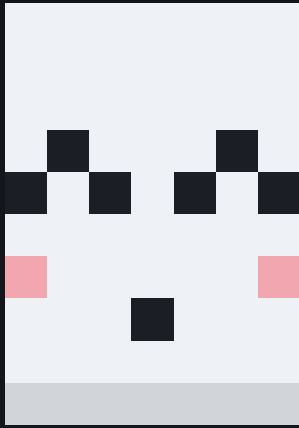
The Bestiary BUILT

The Glossary and the Ledger BUILT

Contents

- Opening 5
- 1** Building Your Lab 6
- 2** The Web Labs 10
- 3** Network and AD Labs 13
- 4** Cloud and Container Labs 16
- 5** The Capstone 19
- Worked answers 24

A place to break things safely



You had read enough of the library to itch to try it, and Blink was ready to point something live and see what happened.

BLINK "Let us try it on something real. Right now."

SOL "On something you own. Everything in this library is practice, and practice has one unbreakable rule, the same one the whole shelf is built on: you test what you own, or what you were invited in writing to test, and nothing else, ever. So first you build a lab that is yours, sealed off, safe to wreck. Then you break it on purpose, as many times as you like."

Worm's eye was on the one thing that made the practice clean: the boundary of the owned box.

WYRM "your box. or paper."

So the book is the workbook: build the lab, then work the web, the network, the cloud, and a full capstone engagement, every target one you own or are lawfully invited to touch.

CHAPTER ONE

Building Your Lab

Every other volume in this Library hands you a technique and says practise it. This chapter builds the place where practising is safe, and sets the one rule that outranks every technique in the building.

You learn this craft by being wrong a great many times, quickly, on purpose. A lab is the room where being wrong costs nothing: no production to break, no stranger to harm, no law to answer to. Volume II builds your first web lab in passing. This workbook is the volume that builds the whole range and exercises every other shelf against it, so the first job is to raise a space that is yours and sealed, and to agree the rule that governs everything you point a tool at.

The rule comes before the build

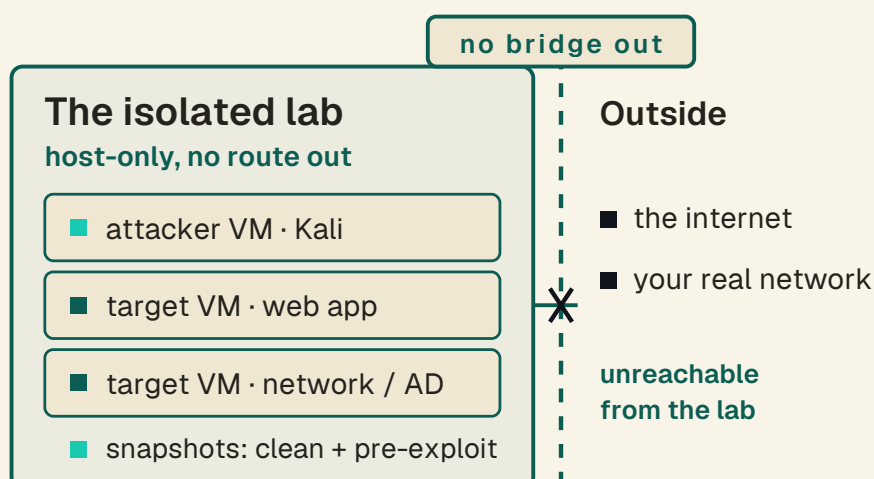
There is one sentence under all of this, and it is not a technique. You test only what you own, or what you have been invited in writing to test. Everything in this workbook is built to run on targets that are yours: a VM on your own disk, a cloud account you pay for, a practice platform that exists to be attacked and says so in its terms. The law that sits behind the sentence is Volume 0, in full, and it is not re-taught here. What is taught here is the habit, and the habit is set on day one, in your own lab, so that it is already automatic the first time you sit in front of a client's system.

Four pieces, one seal

A working lab is small. An attacker machine (Kali Linux is the usual choice, because the tools are already on it). One or more target machines that are deliberately weak. A hypervisor to run them, such as VirtualBox, which is free. And a network layout that seals the weak machines off from everything real: your home network, your work network, the internet. The seal is the part people skip and the part that matters most, because a deliberately vulnerable VM on your real LAN is a deliberately vulnerable VM that anyone on your real LAN can reach.

Pick the right network mode, on purpose

A hypervisor gives a VM several ways to connect, and the names mislead beginners. Bridged puts the VM straight onto your physical network, with its own address on your LAN, reachable by every device in the house. You do not want that for a target. NAT lets the VM reach the internet but does not expose it inbound. Host-only joins the VMs and the host to each other on a private segment with no route out. Internal network joins the VMs only to each other, not even the host. For the vulnerable segment, host-only or internal is the answer. The attacker box can hold a second adapter you bring up only when you deliberately need to fetch an update, then take back down.



A sealed lab: host-only, nothing bridged out.

■ EXERCISE: RAISE THE RANGE

Install a hypervisor (VirtualBox is fine and free). Create a Kali VM from the official image and give it two adapters: one host-only, one NAT left disconnected by default. Put the host-only adapter on a dedicated network, for example `192.168.56.0/24`, VirtualBox's default host-only range. Boot Kali, confirm it has an address on that segment, and take your first snapshot while it is clean. You now have an attacker machine and a sealed segment, and a clean state you can always return to. Write down, in one line, which adapter is which, because the single most expensive mistake in this chapter is attacking on the interface you thought was isolated and was not.

Isolation is a claim you verify, not assume

You do not get to believe the segment is sealed because you chose the host-only option in a menu. You believe it because you checked from inside. From a target on that segment, the default route should go nowhere useful and a ping to a public address should fail. That is a thirty second check and it is the difference between a lab and an accident.

■ VERIFY IT

Boot a target on the host-only segment. On it, run `ip route` and read it: there should be no default route reaching the internet. Then run `ping -c 2 1.1.1.1` and confirm it fails, and `ping -c 2` the address of your home router and confirm that fails too. Now ping your Kali box on the segment and confirm that succeeds. Two failures and one success is the shape you want: the target can talk to the attacker and to nothing else. If any public ping succeeds, stop and fix the network before you run a single tool, because the target is not sealed.

■ EXERCISE: THE UPDATE LANE

Targets stay offline. The attacker box sometimes needs packages. Practise the controlled lane now, while nothing is at stake: with the vulnerable targets powered off, enable Kali's NAT adapter, run your update, then disable the NAT adapter again and power the targets back on. The point is to make reaching the internet a deliberate act with the weak machines absent, never an always-on default with them present. Do it twice so the sequence is muscle memory.

Snapshots are your undo

A snapshot freezes a VM's whole state so you can return to it exactly. Take one when a machine is clean and freshly built, and another before you do anything destructive. When an exploit corrupts the box, or you want to try a second path from the same starting point, you roll back in seconds instead of rebuilding for an hour. Snapshot early, snapshot before each big step, and name them so you know what each one holds.

■ NOTE

Resource it honestly. Each VM wants its own memory and disk, and an AD lab later in this workbook wants several at once. Check what your machine can actually spare before you build the bigger ranges, and run only the VMs a given lab needs. A lab you cannot boot teaches nothing.

■ THE LINE

Own it or be invited. Everything you attack in this workbook is a VM on your disk, a cloud account you pay for, or a platform that exists to be attacked and says so in writing. Never a machine because it was reachable, never a site because you have an account on it, never your employer's network because you were curious. The law is Volume 0 and it does not bend for a lab that happens to be convenient. Build the range so that the only things in reach are things you are allowed to break, and the rule enforces itself.



SOL SAYS

The discipline you build in this empty room is the discipline you carry into a client's. Nobody watches you set up your own lab, which is exactly why it is the honest test of the habit. If you will seal the network and verify the seal when there is no one to impress and nothing yet to find, you will do it on the day it is a real network and a real client, and that day is when it counts.

**not bridged.**

The quick way to update a stubborn target is to flip it to bridged for a minute and let it onto the real network. For that minute a machine you built to be broken is sitting on the same LAN as everything you own. Bring the attacker box out through its own lane instead, with the targets powered down. The shortcut is the whole risk.

■ ON YOUR FEET

A friend messages you: their small business site feels slow and they think someone is poking at it. They give you the admin password and say, "you know this stuff, just have a look, you have my permission." You have the skills and they have asked you. Based on this chapter, what is wrong with going ahead, and what do you put in place before you touch anything?



WHAT WYRM KEEPS

A lab is an attacker box, targets you own, a hypervisor, and a network sealed from everything real. The rule that owns it is older than any tool: own it, or be invited in writing.

- Host-only or internal networking for the vulnerable segment; never bridged. The attacker box reaches the internet through a separate lane, brought up only on purpose with the targets powered down.
- Isolation is a claim you verify from inside the target: no route out, public pings fail, only the attacker is reachable. Checked, not assumed.
- Snapshot a clean build and snapshot before each destructive step, so being wrong costs seconds, not an hour.
- The legal and ethical ground rule is set here, on your own machines, so it is already a reflex when the system belongs to someone else. The law itself is Volume O.

CHAPTER TWO

The Web Labs

Three deliberately vulnerable web targets, and a job for each. By the end you have reproduced the main classes from Volume II with your own hands, and seen why the fix works by reading the code at each level.

Volume II taught the web vulnerability classes and the method for finding them. This chapter is where you put your hands on them. Three targets carry the work, each chosen for a different reason: one gives you a difficulty dial and its own source to read, one is a modern single-page app with a scoreboard, and one is a hosted set of single-bug labs that reach into the harder classes from The Web, Deeper. You run the first two on your own machine; the third you practise on its owner's infrastructure, because they invite you to.

Three targets, three jobs

DVWA, the Damn Vulnerable Web Application, is a small PHP app with a security level you can set to Low, Medium, High, or Impossible. The gift is not the bug, it is that you can read the source behind each level and watch the same vulnerability get harder to reach and then close. OWASP Juice Shop is a modern JavaScript application with a built-in scoreboard that tracks the challenges you solve, which makes it a good map of a real app's surface. PortSwigger's Web Security Academy is a free online set of labs, each isolating one vulnerability, hosted by PortSwigger for exactly this purpose, and its later labs cover classes that live in The Web, Deeper.

DVWA: turn the dial, read the source

Run DVWA locally, log in, and set the level to Low. Take the injection you learned in Volume II and reproduce it until a response shows you data it should not. Then set the level to Medium and try the same input: it will fail or need reshaping. Now open the source for that module at both levels, side by side, and read exactly what changed. That reading is the lesson. A payload that works is a party trick; knowing which line of code stopped it, and why, is the skill the client pays for.

■ EXERCISE: THE DIFFICULTY DIAL

On your own DVWA instance, pick the SQL injection module. At Low, retrieve a row you were not meant to see and note the exact input. At Medium, reproduce it or explain in one sentence why your Low payload no longer reaches. At High, do the same. Then read the module's source at all three levels and write, for each step up, the change that raised the bar, and whether it closed the hole or only moved it. Finish at Impossible and name why that version cannot be reached at all. You are mapping the class from Volume II onto real code, from working to closed.