



GHOST
PROTOCOL



THE FIELD LIBRARY

OPERATIONS

INSIDE THE SCOPE

Bug Bounty

What bug bounty is, reading a programme,
finding and triage, the report that pays, and
the long game.



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

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

THE GHOST PROTOCOL FIELD LIBRARY · OPERATIONS

Bug Bounty

What bug bounty is, reading a programme,
finding and triage, the report that pays, and the
long game.

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 YOU ARE HERE

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 BUILT

Case Files BUILT

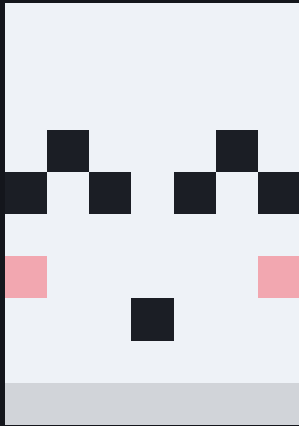
The Bestiary BUILT

The Glossary and the Ledger BUILT

Contents

- Opening 5
- 1** What Bug Bounty Is 6
- 2** Reading a Programme 10
- 3** Finding and Triage 15
- 4** The Report That Pays 19
- 5** The Long Game 23
- Worked answers 28

Inside the scope



Blink had found the bounty boards and was already dreaming of a payout, scrolling programmes with a hungry cursor.

BLINK "Open targets. Real money. I just start hunting, yes?"

SOL "You start by reading. A bounty programme is a contract written as a scope: these assets, these kinds of bugs, these rules. Inside that boundary you are invited. One step outside it and you are not a hunter, you are trespassing, and the same act that pays inside the scope is a crime outside it. The scope is not the fine print. It is the whole of it."

Wyrms eye settled on the out-of-scope list, the part most people skip.

WYRM "read that first."

So the book is how to earn from this cleanly: read a programme before you touch it, find what is actually in scope, triage honestly, write the report that pays, and play the long game where your name is worth more than any single bounty.

CHAPTER ONE

What Bug Bounty Is

A company invites strangers to attack its own systems, inside written rules, and pays for the real problems they find. This chapter is the model, and the two separate permissions you need before you touch anything.

Bug bounty is the one corner of this work where someone you have never met publishes an open invitation to test their systems and promises to pay you for the security bugs you report. It is real, it is large, and it is one of the few ways a tester outside a firm can get legitimate practice on production software. It is also the easiest place to drift across a line and turn practice into a criminal offence, because the invitation is never as wide as it first looks.

The shape of the deal

There are three parties. The **organisation** owns the systems and runs a programme: it writes a policy that says what you may test, how you may test it, and what it will pay. The **platform** sits in the middle (HackerOne, Bugcrowd, Intigriti, YesWeHack and the rest): it hosts the policy, takes your reports, runs triage, tracks your reputation, and moves the money. You are the **researcher**: you read the policy, test only what it names, and report what you find in the shape it asks for.

The economics are simple and unforgiving. A reward is paid for a finding that is valid (it is a real security issue), unique (you are the first to report it), and in scope (the policy covers the asset and the technique). The first correct report of a given bug wins; the second gets nothing, however good it is. There is no retainer, no guaranteed hours, no salary. You are paid per result or not at all, and for a long time at the start it is mostly not at all. Set that expectation now and the work will not surprise you.

VDP and paid bounty are not the same thing

Two kinds of programme wear similar clothes, and confusing them wastes weekends. A **vulnerability disclosure programme**, a VDP, is an open door for good-faith reports: a safe, named way to tell an organisation it has a bug, usually with public credit and a promise not to pursue you legally, but no money. A **paid bounty** does all of that and pays a reward for each valid, unique, in-scope finding, with the size of the reward scaled to severity. Many organisations run a VDP first and add bounties to the assets they most want tested later. Which one you are in is a fact you read off the policy page, never a thing you assume because the logo looks like it can afford to pay.

Programmes are also either public or private. A public programme anyone may join and start reading. A private programme is invite-only, and the invites go to researchers who have already built a track record, usually on that platform. Private programmes are quieter, less picked-over, and less prone to the duplicate problem, which is why reputation is worth building even before it pays directly. More on that in Chapter Five.

The platforms, in one breath

You do not need to choose cleverly on day one, so here they are in brief. **HackerOne** and **Bugcrowd** are the two big US-based platforms, each hosting a large number of corporate programmes (HackerOne counts more than a thousand active ones). **Intigriti** (Belgium) and **YesWeHack** (France) are the main European platforms. **Synack** is vetted: you pass a background check and a skills assessment before you see a target, and the crowd is small and run more like a managed workforce. **Immunefi** is the platform for web3 (smart-contract) programmes. Pick based on where your skills fit and which policies you can actually read and respect, not on the name above the page.

One thing about the state of these platforms matters for how you work. Since around 2025 the large public programmes have been flooded with low-quality, machine-generated reports, so valid-report rates are low and triage queues are slow, and some well-known maintainers have publicly scaled back or left over the noise: the curl project ended its HackerOne bounty at the end of January 2026, citing the flood of low-quality reports. The practical lesson for you is the opposite of discouraging: a clear, verified, genuinely-reproduced report now stands out more than it ever did, because so much of what lands in the queue beside it does not.

Two authorisations, not one

This is the sentence the whole volume is built to make a reflex, and it is the same shape as the rule in Volume 0. A bounty programme is a written authorisation, scoped exactly like the Rules of Engagement in a client test. That is the first lock, and it is real: the policy page is a contract that says, in writing, what you are allowed to do. For a Ghost Protocol intern there is a second lock, and it is not optional. During your internship you do not take on bug bounty, or any outside testing, without clearing it with your supervisor first, in writing.

The second lock is not the firm being controlling. It protects you, by keeping you from drifting over a programme's scope line and into an offence under the Computer Crime Act (Volume 0). It protects Ghost Protocol, because your conduct carries the firm's name whether you intend it to or not. And it avoids a mess, because the asset you fancy on a Saturday might belong to a GP client, or might quietly eat the hours GP is already paying you for. Volume V states this same rule from the other side, in the context of staying sharp; this volume is where you learn to work inside it in detail.



two.

The programme's written scope is one. Your supervisor's written yes under the internship terms is the other. Missing either, and the answer is the word Wyrn has said since Volume 0.

Scope is the whole game

Everything else in this volume is a footnote to one idea. The programme's scope is the edge of your permission, and nothing past that edge is yours to touch, no matter how easy it is to reach, how close it sits, or how plainly the same company owns it. The skill to reach a thing is still not the right to reach it; that was the one rule in Volume 0, and a bounty does not suspend it. An asset that belongs to the

client but is not listed in scope is exactly the `admin.client.com` -next-door case from Volume II, Chapter One: you record it, you may mention it to the programme if the policy invites that, and you do not test it. A programme can only authorise you for what the programme controls, and only for what it has actually written down.

■ THE LINE

Before your first request against any bounty target, three answers must all be yes, out loud, from the policy in front of you and your supervisor's written reply:

- Is this exact asset named in the programme's in-scope list, and is my technique one the policy permits?
- Have I cleared outside testing with my GP supervisor, in writing, under the internship terms?
- Are both of those current, specific, and still true today (not a yes from six months and three scope changes ago)?

Three yeses, or you do not touch it. "It has a bounty programme" is not one of the three.



SOL SAYS

A bounty is practice that can pay. It is not a licence, and it is not a loophole around the oath you signed in Volume O. The policy page is an ROE written by a stranger who has never met you and will never trust you more than your first out-of-scope request lets them. Read it as carefully as you would read a contract you were signing, because that is what it is.

Where this sits in the Library

This volume does not re-teach the things other volumes own. The law, the one rule, and clean conduct are Volume O, and they govern every page here. The web vulnerability classes you will be hunting are Volume II, taught as mechanisms; this volume assumes you know them. The report you send a triager is a close cousin of the Ghost Protocol engagement report in Volume IV, and Chapter Four shows how they relate. And the discipline of sharpening on sanctioned ranges, with bug bounty treated as the double-locked door it is, is Volume V. Read this volume as the working detail of a rule those volumes already set.

■ EXERCISE

Open the public programme directory on HackerOne or Bugcrowd and pick one programme whose subject you understand. Reading only (this is a scope page, not a target), write down five things: who the organisation is, whether a platform mediates it, whether it is a VDP or a paid bounty, whether it is public or private, and one asset that is clearly in scope and one that is clearly out. If you cannot answer all five from the policy page alone, you have not finished reading it. You will do the deeper version of this read in Chapter Two.

■ VERIFY IT

Three claims look true and are the ones most worth checking before you spend time: "this company has a bounty programme", "this asset is in scope", and "this bug pays". Prove each from the policy page itself, not from a blog post, a tweet, a conference talk, or a wildcard you have read generously. A company having a security.txt or a years-old VDP does not mean the host in front of you is in a paid programme today. If the policy page does not say it, you have not verified it, and you proceed as if the answer is no.



WHAT WYRM KEEPS

Bug bounty is a written invitation to test named systems for pay. The invitation is only ever as wide as the policy page. Two authorisations, not one.

- Three parties: the organisation runs the programme, the platform mediates, you read and obey the policy. Paid per valid, unique, in-scope finding, or not at all.
- A VDP gives safe-harbour and credit, not money; a paid bounty pays by severity. Programmes are public or private. Which one, and whether it pays, is read off the policy, never assumed.
- A bounty programme is an ROE (first lock). The GP internship terms require your supervisor's written yes to do outside testing at all (second lock). Both, current, or you stop.
- Scope is the whole game. An out-of-scope asset of the same company is the next-door case from Volume II: recorded, not touched.

CHAPTER TWO

Reading a Programme

The policy page is the contract. This chapter teaches you to read it the way you read a Rules of Engagement: what is in, what is out, what is forbidden, what safe-harbour does and does not give you, and how to choose a programme worth your time.

A programme policy is an ROE written by someone who will never meet you. Everything you are allowed to do is in it, and everything you are not allowed to do is either in it or outside it by default. You read it slowly, once, before you touch anything, and again whenever it changes, because it does change and your old reading expires when it does.

What a policy actually contains

The wording differs by platform, but every serious policy has the same parts, and they map almost one to one onto the Rules of Engagement from Volume O. Learn the parts and you can read any platform's version of them.

- **In-scope assets.** The systems you may test, named: domains, specific applications, mobile apps, API hosts, sometimes an IP range. This is the whole of your permission. If it is not on this list, it is not yours.
- **Out-of-scope assets.** Systems explicitly ruled out, often ones the company owns and you might assume were fair game: marketing sites, third-party services, acquired brands, staging and development hosts. The list is not exhaustive, which is the trap: out-of-scope is the default, and this section only names the ones people most often get wrong.
- **Accepted and excluded vulnerability types.** Which classes count. Many programmes exclude whole categories as known or low-value: self-XSS, missing security headers, clickjacking on pages with no sensitive action, reports generated only by an automated scanner, best-practice complaints with no demonstrated impact. Reporting an excluded type is not a finding, it is noise, and it costs you standing.
- **Rules of engagement.** How you may test. Common hard limits: no denial of service, no automated scanning or heavy traffic, no social engineering, no physical intrusion, no testing against real users or their data, use the named test accounts, stay under a rate limit, do not pivot into systems the in-scope app merely talks to.
- **Data handling.** What to do if you reach real data: take the minimum that proves the issue, do not download or retain it, redact it in the report, and stop and report serious exposure rather than exploring it. This is the Volume O data-handling rule, restated by the programme.
- **Disclosure policy.** Whether, when, and how you may talk about a finding publicly. The default on most programmes is coordinated: nothing public until the issue is fixed and the programme agrees. Breaking this can void a reward and worse.
- **Rewards and severity.** How findings are rated (usually a severity scale, often CVSS-derived) and the reward bands that follow. Severity is argued from evidence, which is Volume IV's subject; the policy just tells you how this programme maps severity to a reward.