GUIDE GS-038 / SHELF F / TOOLS & SYSTEMS

Find Offsets After a Patch, Step by Step

Rebuild a broken table in under an hour, in a fixed order.

GUIDE	VERSION	DATE	PRICE
GS-038	v1.0	2026-11-24	$12

WHAT THIS GETS YOU

Turn a game patch from a wasted evening into a routine twenty-minute rebuild.

Patches move code without changing behaviour, which means the value is still there and the path to it moved. This guide is the fixed order that finds the new path fastest.

Written for someone starting from zero. Every term is defined the first time it appears. Read the plain-words section before the steps.

If You Only Remember One Thing

THE ONE THING

The value never moves far. The path to it does. Rebuild the path, not the research.

WHAT YOU GET

- Confirm a patch actually moved the offsets.
- Recover a value with a fresh signature rather than a new scan.
- Turn a rebuild into a written checklist.

WHAT YOU NEED

- Your previous table with verified pointers.
- Cheat Engine with the game attached.
- The old version number and the new one.
- A dated folder for signature files.

HOW TO READ THIS

Steps one to five are diagnosis. Steps six to ten are the rebuild. If diagnosis is skipped, the rebuild starts from zero every time.

Pairs with: GS-037 (Cheat Engine basics), GS-043 (VM lab), GS-042 (proxies).

Plain Words First

OFFSETS VERSUS SIGNATURES

An offset is a fixed distance that changes with every patch. A signature is a byte pattern that follows the code.

A signature survives layout changes; an offset does not.

WHAT PATCHES ACTUALLY CHANGE

Small patches rebuild code sections and shift everything after them. The logic is usually identical.

That is why a signature found last patch usually still works.

PAUSE HERE

If the primer is not clear, reread it. The steps assume these words already mean something to you.

Execution Checklist

Ten steps, one result each. Tick a box only when you can see the result the step promises.

- [ ] **01** — Write the old state down
- [ ] **02** — Confirm the executable changed
- [ ] **03** — Back up the old table
- [ ] **04** — Launch and reach a stable screen
- [ ] **05** — Re-scan for the value
- [ ] **06** — Freeze-test the new address
- [ ] **07** — Find what writes again
- [ ] **08** — Validate the signature uniqueness
- [ ] **09** — Rebuild the pointer and test
- [ ] **10** — Write the rebuild checklist

> IF A STEP STALLS
>
> Stop. Do not push past a step whose result you cannot see. Go to the failure table at the back and fix the cause first.

> IF YOU ARE SHORT ON TIME
>
> The first four steps decide whether the rest work. Do those today and the remainder tomorrow. A half-finished guide that skipped the foundation is worse than no guide.

Steps 01–04

01 Write the old state down

Note the game version, the pointer, and the value it held.

> Include the date the table last worked.
>
> This is your rollback reference.

Why it works: You cannot measure a change without a baseline.

02 Confirm the executable changed

Compare the file timestamp and size.

> A patch that only changes assets leaves code offsets intact.
>
> Check before spending an hour rebuilding.

Why it works: Half of "patch day" problems are not code changes.

03 Back up the old table

Copy the table and label it with the old version.

> Never overwrite the last known-good copy.
>
> Keep it in a dated folder.

Why it works: Rollback is cheaper than repair.

04 Launch and reach a stable screen

Attach after the game has fully loaded.

> Loading screens move memory around.
>
> Attach on the same screen you used last time.

Why it works: Consistent conditions make the comparison valid.

Figure 1 — Old state

```
version: v1.3.0
value: health
offset: 0x01F4
table: health-v1.3.0.ct
```

Steps 05–07

05 Re-scan for the value

Use the same type and the same starting value.

> If the scan finds the value easily, only the path broke.
>
> If it finds nothing, the type or the feature changed.

Why it works: This one step decides how long the rebuild takes.

06 Freeze-test the new address

Confirm the value is live before continuing.

> Same method as the first time.
>
> One frozen address is enough.

Why it works: Rebuilding on an unverified address wastes the whole session.

07 Find what writes again

Capture the instruction and its module offset.

> Note the bytes, not just the address.
>
> The bytes are the signature.

Why it works: The instruction is the stable part.

Figure 2 — Signature capture

```
mov [rax+000001F4],ecx
bytes: 48 89 88 F4 01 00 00
hits: 1
module offset: 0x00000210
```

Steps 08–10

08 Validate the signature uniqueness

Confirm the byte pattern appears exactly once in the module.

> Add bytes until only one match remains.
>
> Too short a pattern matches dozens of places.

Why it works: A unique signature is what makes the next patch easy.

09 Rebuild the pointer and test

Apply the new offsets and verify the value moves with the game.

> Restart the game and confirm the pointer resolves.
>
> Keep the old entry disabled until the new one passes.

Why it works: One verified path beats ten candidates.

10 Write the rebuild checklist

Record the exact sequence that worked, under fifteen lines.

> Include menu names and the signature file name.
>
> Store it with the table.

Why it works: The second rebuild should take a third of the time.

Figure 3 — Rebuild log

```
value: health
new pointer: base+0x210
restarts verified: 2
status: VERIFIED
```

Words You Will See

Offset — A distance from a base address to a value.
Signature — A byte pattern used to find code after it moves.
Module offset — The distance from the module base to an instruction.
Pointer scan — A tool that finds paths from a static base to a dynamic address.
Rebuild — Restoring a working table after a patch.
Uniqueness — A signature matching exactly one location.

KEEP THIS PAGE

Come back when a step uses a word you do not recognise. The definition is here, not scattered through the steps.

What Can Go Wrong

FAILURE	WHY IT HAPPENS	FIX
Value not found at all	Type changed or feature removed	Try other types, then confirm the feature still exists.
Signature matches many places	Pattern too short	Add bytes until one match remains.
Pointer breaks after restart	Path incomplete	Pointer scan and rescan twice.
Table works once	Second process owns the value	Attach to the process with the highest memory.
Crashes on attach	Debugger conflict	Use VEH mode.
Next patch breaks everything	No signature saved	Always store the bytes with the version.

SUMMARY CARD — GS-038

```
1 record old version + pointer 2 confirm executable changed 3 back up the table 4 attach on the same screen 5 re-scan the value 6 freeze test 7 capture writer instruction 8 check signature uniqueness 9 rebuild pointer 10 restart and verify 11 check the other entries 12 write the rebuild checklist
```

VERSION NOTE

Rebuild with a new date when a step stops working. Buyers get the updated PDF from the same order link.