

CodeSeal and the history behind your source code

Think of it like a VIN lookup for a source snapshot: a way to follow recorded history, even when the person sealing the code did not write it.



By Lyra

Published Oct 18, 2026 · 5 min read



A car's VIN gives you somewhere to start when you want to understand its history. You can use it to look up records about that vehicle. Knowing the number does not put your name on the title.

That is the idea behind [CodeSeal](#): give a source snapshot a reference you can look up, then use recorded observations to understand its history. The interesting question is what happened to this code, and when someone recorded it.

Software moves around. We download a public project, make a small change, hand a ZIP to a colleague, and revisit it months later. By then, “the version we used” can mean several different things. A signed observation gives that conversation a concrete starting point.

| The VIN analogy has a useful boundary

Think of the CodeSeal identifier as a handle for finding a record. The fingerprint describes the selected source snapshot, and the registry's digital signature lets you check the integrity of its signed statement.

Those pieces do different jobs. An identifier helps you find something; a fingerprint helps you compare it; a signature helps you check that the signed evidence has not changed. None independently tells you who originally wrote the application.

The analogy has limits, too. A changed snapshot gets another recorded observation. You should not assume every fork, renamed repository, or later version automatically shares one permanent application identity. Read the identifiers and comparison results attached to the actual records.

| Yes, you can seal a public repository

Suppose you download an open-source project and register that snapshot under your account. The useful statement is: this snapshot was submitted and recorded under that identity at that time.

The upstream authors still have their own history, credits, and licence terms. Your registration does not turn their work into yours. It also does not independently prove that you ran the application, wrote its code, or were its first user. The signer details describe the declaration attached to the event.

Now imagine you change a function for your own workflow and seal the resulting snapshot. You have two observations to investigate: the earlier source you registered and the later source containing your change. Keep the upstream revision and your own commit references with your project notes so the comparison has context.

That is a useful record even when authorship was never in dispute. You may simply want to answer, "Which version did we hand over?"

| What happens during a scan

You provide a source ZIP or use a supported GitHub import. CodeSeal inspects source without executing the submitted application. Its file-selection rules and marked private sections define what enters the scan, so the resulting record describes that selected material.

The scanner calculates file fingerprints and combines them into a package fingerprint. It also records a structural fingerprint and normalized block fingerprints that help identify source overlap. Common helpers and recognized dependency or vendor material receive less independent provenance weight.

An exact snapshot match and a block-level similarity result answer different questions. Similarity is a lead for understanding a relationship; it cannot, by itself, explain why two projects share code.

Scanning prepares the evidence. Registration adds the project and signer details to a recorded observation, which CodeSeal signs with its registry key. The recorded time is the registry observation time, rather than a claim about when the code was originally written. Earlier observations remain available for historical comparison.

| Keep the record with the version

Here is a small workflow worth trying with a public project you are allowed to use:

1. Note the upstream URL, exact revision, and licence before making changes.
2. Scan and register that snapshot, then keep its identifier and signed evidence with your project notes.
3. Make a deliberate change and commit it in your own repository.
4. Scan and register the changed snapshot, then inspect the recorded history and comparisons.

Give each snapshot a clear label. “Before our export change” will mean more to a teammate than “final-v3-really-final.” Include the relevant CodeSeal reference in the handoff or release notes so the next person can find the same observation.

Keep the source version yourself, too. CodeSeal's durable evidence is not a backup of the application. File selection and private-section exclusions also matter when reproducing a fingerprint: a record cannot describe material outside its stated scope.

| Larger scans follow the same idea

Large source trees can exceed an upload or import limit. The incremental path now activates when that size limit is identified; ordinary scans retain their existing path.

In that fallback, source is processed one file at a time. Each temporary server copy is deleted after its evidence is prepared, before the file is acknowledged. Your original local files remain in place. This still transfers source for processing, and the published safety limits continue to apply. The final registration concerns the assembled snapshot rather than a separate ownership claim for each file.

| History can help when bigger questions arise

A technical chronology can be useful during a handoff, an intellectual-property review, or a later disagreement about changes. Signed observations may help establish which selected material was recorded at particular times, alongside repository history, licences, contracts, and other evidence.

The broader ownership question still needs that broader record. CodeSeal does not independently establish authorship, copyright, permission to reuse code, or legal title. Its [legal notice](#) sets out that boundary explicitly.

That is how I would approach the value on the screen: use it to find the observation, inspect what was included, and compare it with the version you have. Like a VIN lookup, its usefulness comes from the history you can investigate through it. Start with a snapshot you understand, keep the evidence beside that version, and make the next change easier to explain.

| Sources & further reading

Checked 2026-10-18 · Story context 2026-10-18

1. [CodeSeal scanning and provenance workflow](#) (2026-10-18)
2. [CodeSeal technical provenance and ownership boundaries](#) (2026-10-18)
3. [CodeSeal source and metadata privacy](#) (2026-10-18)

[Read the current article online →](#)

Support the workshop

Explore software and learning offers at [StackSocial](#). Affiliate link: ProgreTech may earn a commission from purchases. Check current price, license and renewal terms before buying.

Find our projects at [progretech.com](#).

ProgreTech LLC · Build / Learn / Improve / Together

This file contains its text, illustrations and fonts. No account, scripts or network connection required to read. External reference links need internet access.