Replication Certification at Publication: Checkable by Anyone, Anywhere, in Seconds, Forever*

Gary King† Yiqing Xu‡ Leo Y. Yang§

November 2, 2026

Abstract

Authors increasingly deposit replication datasets—the data and code behind publications—in archives, and journals increasingly require them to, under progressively onerous rules. But a deposit alone proves little, so authors keep paying: months for the journal's reanalyses, years answering readers' questions, and rancor as failed replications become accusations because no record establishes what actually ran. We introduce a replication certificate: a cryptographically signed record of the deposited files and logged output. It replaces the journal's reanalyses with evidence anyone can check in seconds, which no one, including the archive, can backdate, and no one can fake without risking detection. Certification asks little of anyone and benefits everyone: The author deposits, then reruns the analysis once under a random seed the archive computes from the deposit's hash and a blockchain block; the archive signs, timestamps (in that blockchain), and stores the record—confidential data included—without rerunning anything; the journal merely checks a website for the certificate; and anyone can check or replay any step decades later.

*We thank Danny Ebanks, Stefano Iacus, Chris Lucas, Katalina Toth, Ana Trisovic, and Salil Vadhan for helpful comments and suggestions. The present version of this paper and the Supplementary Appendix are available at GaryKing.org/certify.

†Institute for Quantitative Social Science, Harvard University; GaryKing.org, King@Harvard.edu.

‡Department of Political Science, Stanford University; yiqingxu.org, yiqingxu@stanford.edu.

§Stanford Center on China's Economy and Institutions, Stanford University, leoyang@stanford.edu; currently affiliated with the Department of Accountancy, Economics and Finance, Hong Kong Baptist University; LeoYang.org, leoyang@hkbu.edu.hk.

1. Introduction

Ensuring that evidence offered in a scholarly publication stands independently of its author is a goal that echoes through the history of science, back to the 1660 motto of the Royal Society, *Nullius in verba* ("take nobody's word for it") and even to Heraclitus and Kālāma Sutta more than two thousand years earlier. We can also see this goal in the now widely accepted "*replication standard* [which] holds that sufficient information exists with which to understand, evaluate, and build upon a prior work if a third party could replicate the results without any additional information from the author," (King, 1995, p.444), and in many efforts to meet the standard.[1]

Scholarly publication outlets over four decades have adopted increasingly strict policies designed to convince authors to deposit *replication datasets*, the data and code behind a published result, in professional *archives*, the data repositories such as Dataverse, that store, version, and cite them, and to try to ensure each deposit provides meaningful evidence that results can be reproduced (Section S8 describes everything we require of them). Unfortunately, each of the policy changes has merely moved the burden of verifying replicability from one actor to another, while often increasing costs in money, staffing, publication delays, time investment by authors and future researchers, and misunderstandings between them, while still never quite succeeding. Section 2 explains.

Instead of even more onerous and expensive policies, we propose a technological solution to accomplish the same goals more effectively and at lower cost. By marshalling well-tested cryptographic methods, we show how to create a special replication dataset *certificate* that provides the evidence that the analysis ran on exactly its files and produced exactly the results it records, with its random draws made after the files were fixed. The journal can then substitute its expensive and time-consuming in-house replication runs with a quick check that the certificate exists before final acceptance, and a later reader can check the record in seconds instead of re-running the analysis. A failed replication then has a certified

[1] We define "replication" as most users and data repositories do: running the same code on the same data, producing the same result. Some call this "reproduction" and reserve "replication" for a study redone with new data (National Academies of Sciences, Engineering, and Medicine, 2019); we use the words interchangeably.

record to be argued from rather than a reputation to be argued about. See Section 3.

The rest of the paper states what certification does not cover (Section 4) and reviews what our proposal borrows from prior work and what that work leaves open (Section 5). The Supplementary Appendix provides all technical details, including what evading certification costs an attacker (Section S5.2) and our contributions (Section S6).

2. Problems with Replication Policies

No policy tried thus far lets anyone verify a deposited replication dataset without re-running it, so each future researcher has to verify the same deposit from scratch. We describe here the levels of enforcement journals have escalated through; the evidence indicating how these policies have failed; and the costs imposed by the policies.[2]

2.1. Policy Enforcement Levels

Level 1 is the original and least stringent proposal, where journals require authors to "indicate in their first footnote in which public archive they will deposit the information necessary to replicate their numerical results." Editors would be "responsible only for the existence of the footnote, not for confirming that the data set has been submitted nor for checking whether the results actually can be replicated." Any verification "can and should be left to future researchers" (King, 1995).

Level 2 requires authors to go a step further and make data available on request, as in *Science*'s rule from 2011 or by encouraging authors to make data available in some unspecified way. But asked for the materials behind 204 papers published under it, only 44% of authors supplied them, and the results of only 26% of the 204 could be reproduced (Stodden, Seiler and Ma, 2018).

Level 3 makes replication dataset deposit a condition of publication, as now required by many leading outlets (Brodeur et al., 2026; Ferguson et al., 2023; Miske et al., 2026).

[2]The norm that authors deposit the code and data behind a published result is long-established in the social sciences (Dewald, Thursby and Anderson, 1986; King, 1995), and professional archives now supply the storage, versioning, and persistent identifiers this protocol builds on (King, 2007). The Transparency and Openness Promotion guidelines extend the same levels across the sciences, with verified re-execution as their top level (Nosek et al., 2015).

In economics, the requirement reached the *American Economic Review* in 2005 and the *Quarterly Journal of Economics* in 2016 (Christensen and Miguel, 2018). In political science, the editors of twenty-seven journals signed the joint statement of the Data Access and Research Transparency initiative, committing to require that cited data be available in a trusted repository (Lupia and Elman, 2014; Data Access and Research Transparency (DA-RT), 2015). Among quantitative political-science articles, a working link to the data and code was rare before about 2008, with the share growing approximately exponentially since (Rainey et al., 2025, fig.1).

Finally, Level 4 is re-execution of the code by the journal, originally as an option and then a requirement. From July 2009, *Biostatistics* let authors ask it to run their code and marked the article *R* if it reproduced; in two years, five of 125 articles earned the mark (Peng, 2011). Journals now increasingly require human-verified re-execution by the journal's data editor (Vilhuber, 2020; Nüst and Eglen, 2021). The *American Journal of Political Science* has worked with the Odum Institute at the University of North Carolina since 2015 to verify every accepted quantitative article, postponing publication by seventy-five days on average (Jacoby, 2018). New proposals for even more stringent practices also regularly appear (e.g. Reiss et al., 2026).

2.2. Policy Failures

The literature suggests three conclusions. First, *archival deposit alone does not show that the code runs.* Of more than nine thousand R scripts deposited in Dataverse, 74% did not run to completion without error (Trisovic et al., 2022). Of 15,817 Python notebooks from biomedical publications with declared dependencies, 1,203 of those ran through without error (Samuel and Mietchen, 2024), in line with earlier mass executions of notebooks (Pimentel et al., 2019). When two teams each tried to redo one analysis from each of 18 microarray papers, ten could not be reproduced (Ioannidis et al., 2009).

Second, *running the code does not guarantee that the materials reproduce the article's results.* For all available replication packages for articles published 2000–2026 in major economics journals, only 11.2% of 4,452 could be fully reproduced (Schwartz, Andrews and

Shapiro, 2026). Working only from deposited files, one audit reproduced the key result of just 22 of 67 published economics papers (Chang and Li, 2022). At *Cognition*, mandated deposit raised the availability of usable data sharply, while 13 of 35 audited articles still had at least one value that could not be reproduced even with the authors' help (Hardwicke et al., 2018). In the largest journal-run audit, 68% of attempts succeeded in replicating results (Fišar et al., 2024).

And finally, *journal verification catches more failures without catching them all.* Of the twenty-four papers given in-house replication review at the *Quarterly Journal of Political Science*, 14 of 18 produced results that differed from those in the manuscript (Eubank, 2016). Across 254 papers from the three leading political-science journals whose materials could be obtained, 61 did not reproduce every coefficient exactly and 20 did not reproduce it even approximately (Xu and Yang, 2026).

2.3. Policy Costs

Finally, journal replication policies impose three categories of costs.

The first cost is *time*, which comes in two parts, delayed acceptance and endless post-publication questions. In the former, median verification times at the American Economic Association's journals have run three months per round, and some manuscripts need several rounds (Vilhuber, 2024); the average manuscript the *AER* published in 2025 spent thirteen weeks in the data editor's hands, up from nine in 2022 (Luttmer, 2026). In the latter, authors assemble replication datasets to a data editor's standard but still have to answer questions about them for years. Having access to a deposited replication dataset is far better than no deposit at all, but still considerable work: in one mass-reproduction project, teams averaged thirteen days per paper, and about one article in twelve required more than a hundred days (Brodeur et al., 2024). None of that work is recorded where the next reader can use it and so every researcher starts from scratch.

The second cost is *money*, paid by journals (and by editors in their time) which hire and manage replicators, in addition to providing cutting edge computational resources used by researchers. The American Economic Association's Data Editor pays for all student em-

ployment and pre-doctoral staff, while Cornell University's Economics Department provides tuition waivers and computing (Vilhuber, 2025). The Cornell laboratory that replicates received, in one year, 529 requests covering 415 manuscripts. About 15-20 undergraduate research assistants work there per week, supervised by the three staff (Vilhuber et al., 2022). At the *American Journal of Political Science*, verification by the Odum Institute took an average of eight person-hours per manuscript, and authors resubmitted materials 1.7 times on average, a "not trivial" financial cost (Jacoby, Lafferty-Hess and Christian, 2017). *Management Science* charges $79 on every original submission since 2025, to fund its reproducibility infrastructure and external verification (INFORMS, 2025).

The final cost is *rancor*. When a replication fails, the absence of a shared record of what was actually run often turns technical questions into personal ones. The dispute over Reinhart and Rogoff's result turned on what had actually been run—which spreadsheet, which exclusions, which weighting—and was settled only when the authors found and shared the original file (Herndon, Ash and Pollin, 2014). The dispute over LaCour and Green's article turned on whether the deposited data had been collected at all (Broockman, Kalla and Aronow, 2015). Daniel Kahneman, acting as a senior statesman, has gone so far as to propose etiquette rules for how readers attempting replications should approach authors (Kahneman, 2014).

3. A Technological Solution: The Certificate

Figure 1 gives an overview of the certification protocol. The first three parts of this section follow the figure's first three panels.

3.1. How It Works

Certification has three parts corresponding to the three bands within Panel 1 of Figure 1. First, the author makes the replication dataset READY by locking down all sources of variability (due to the code, computing environment, or settings), so that it gives the same results if run repeatedly. Second, the author and archive work together to CERTIFY the replication dataset. And finally, anyone, at any point in the future, can CHECK the results

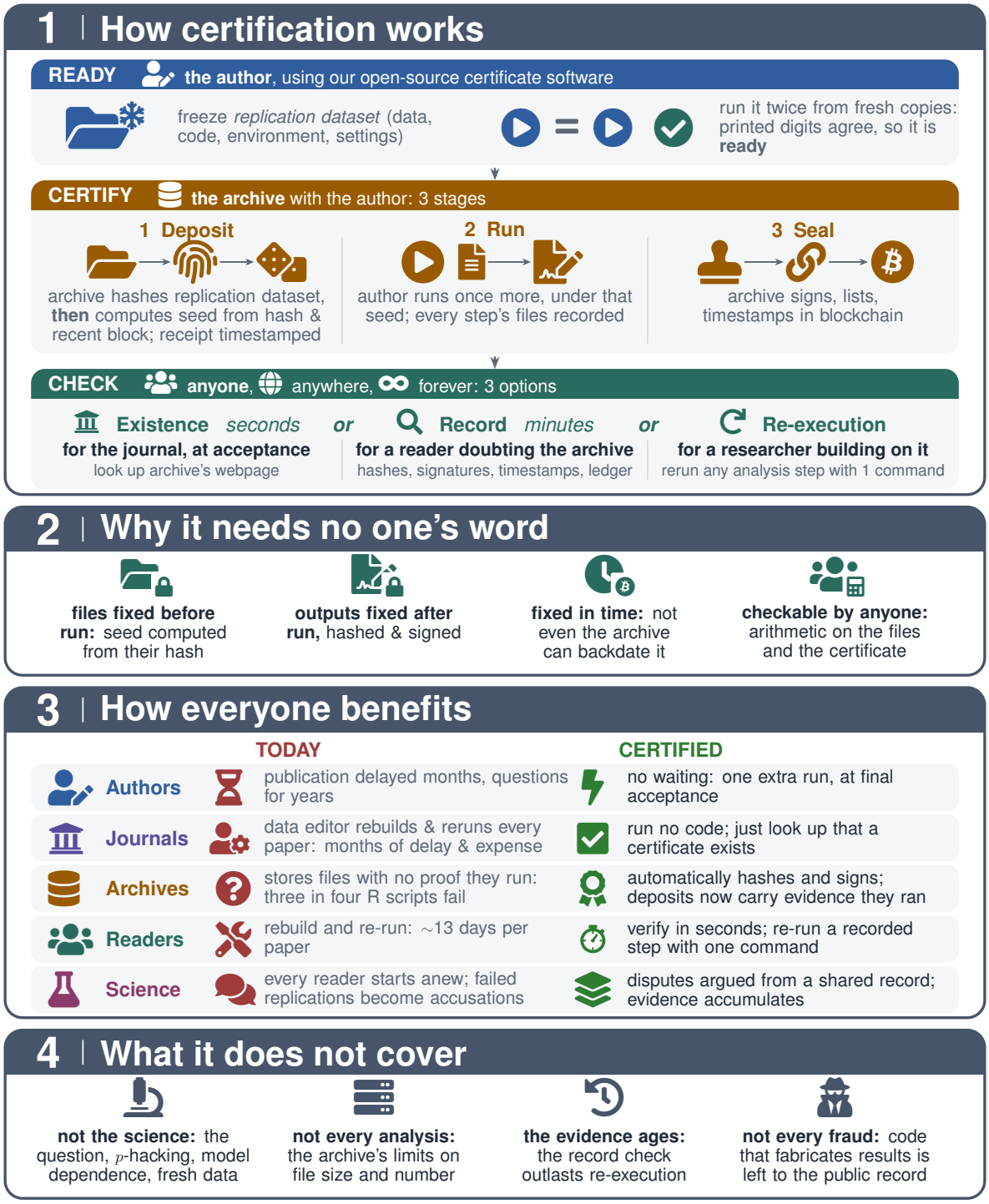


Figure 1. Certification at a glance. Panel 1: READY, the author prepares the replication files until two runs agree. CERTIFY, the archive computes a seed from the deposit's hash and a blockchain block, the author runs the analysis under it, and the archive signs and timestamps the record; draws fall after the seed is computed and before the certificate is issued. CHECK, three options, depending on need. Panel 2: why the record is evidence. Panel 3: what each party pays TODAY and gains once CERTIFIED. Panel 4: what certification does not cover.

in seconds without having to rerun the analysis, and each check catches frauds that pass the one before it.

3.1.1. Ready: Author Prepares the Replication Dataset

We now formalize the current practice of authors preparing replication datasets for deposit (and thus also prior to our certification protocol). Conceptually, getting "ready" in this way involves the author using our certificate software to ensure their replication dataset gives the same results when run at least twice on their machine, which is a minimum definition of "replicability". Authors are advised to conduct more than two runs if results may be fragile, since two runs can agree by luck.

To get ready formally, the software first collects (a) every file in the replication dataset with its hash and the results of a *dependency audit*, a reading of the scripts for the files they open and the seeds they set; (b) the settings to freeze, such as the time zone and the number of processor cores; and (c) the software environment itself (deposited as add-on libraries installed inside the replication dataset, with a "lockfile" that lists their exact versions, or, where software licenses allow, as a "container image", a packaged copy of the operating system and programs). From these, the software creates a *closed copy*, which includes the replication dataset plus a few small files for record keeping. This closed copy enables all the author's analyses to be run with one top-level command, keeps track of each step in the process, and, where possible, has only one input for a random seed.

With the closed copy, the author conducts at least two runs, which we call *closure runs*, each from a fresh temporary copy and under a seed of the author's choosing. The certificate software compares every file the two runs writes and records how closely they agree, as the *determinism tier*: the standard later re-runs must meet. The author cannot declare a tier, only earn one: careful code earns a stricter tier, and unstable code a looser one that the certificate displays. The dataset is *ready* when the two runs agree on every number to a fixed precision, which we recommend setting to at least what articles ordinarily print (such as 2-4 digits), though the goal is byte-identical agreement; the software writes a report, deposited with the replication dataset, that names any file that differed. See Section S2 for details.

3.1.2. Certify: Author Deposits, Receives Seed, Runs Once; Archive Seals

Certification is designed to eliminate the possibility of *backdating* by anyone including even the archive (in order to surreptitiously change the record after the fact), *seed shopping* (where the author cherry-picks a supposedly random seed to make their results look better), *substitution* (replacing an intermediate file, such as the cleaned data, after the run while claiming the certificate still covers it), and *ambiguity* (in what results the run produced or how to conduct a new one). Certification occurs after conditional acceptance by the journal; an author may choose to certify earlier, or to certify a revised replication dataset after publication and link it to the earlier one.

To take a dataset that is "ready" and certify it to receive these benefits requires three stages (corresponding to the Certify row of Panel 1 in Figure 1).

First, the author *deposits* the closed copy in the archive. The archive then computes a new random seed from three inputs: a hash of the uploaded files, a cryptographic timestamp, and the private half of the archive's *seed key*, a key pair it uses only for this purpose, so that no one, the author included, could know the seed before it is computed. Version 1 uses Bitcoin's blockchain for the timestamp for both bounds (other options include any public record that no party controls and no one can backdate, with a public source of values no one can predict in advance). The archive's receipt, its signed and timestamped statement that these files arrived, records the hash, the block, the seed, and a short proof with which anyone holding the seed key's public half can verify that the seed was computed from those inputs.

Second, the author *runs* the closed copy once under the seed issued by the archive (producing the "certified run"). For each step in the run, the certificate software records the "pre-state" (every file and its hash present before the step starts), any files it writes, and any output it produces. Because each step begins exactly where the previous one ended, every pre-state already includes the hashes of all files written by earlier steps, so the record of each step is tied to everything that came before it. The software then combines the records of all steps, in order, into a single hash so that changing any intermediate file, or dropping, adding,

or reordering a step, would change this hash. The certificate therefore vouches not only for the final numbers but for the entire path from the closed copy to the published results. The published numbers must come from this run. The software produces a "proposed certificate".

Finally, the archive *seals* the proposed certificate submitted by the author. To do this, the archive checks the certificate, refuses on any failure, signs the "witness" (its statement that it sealed this certificate, using its signing key), timestamps the certificate in the blockchain, and enters the witness in its ledger (a public list of every document it issues including this one, kept so that any later edit is visible publicly). The archive stores the certificate and records it in a metadata field linked to the published dataset. See Section S3 for details of certification.

3.1.3. Check: Anyone Can Easily Verify the Record

A key advantage of a certificate is that anyone can check it without taking anyone else's word for what was run or when. Three types of checks are available, depending on need; after describing them, we show that each catches frauds that pass the one before it; and finally where a certificate sits among higher levels of assurance. Section S4 provides details.

Three Types of Checks. First, an *existence check* enables a journal to confirm that the replication dataset, the article's citation of it, and the certificate exist (and that the certificate carries no exception mark, the archive's flag indicating that the certificate fails to meet one of the protocol's standards). This requires only a fast lookup on the archive's webpage; no reanalysis is necessary. An existence check is sufficient for the journal because a sealed certificate enables anyone, without help from the journal, to verify both "record consistency" (that the stored files are those the run started from and wrote, and that neither the files nor the record has changed since) and "replication" (that rerunning the recorded steps from their stored starting files produces outputs that agree with the certified ones within the certificate's determinism tier).

Second, a *record check* is designed for someone who prefers not to trust the archive's webpage. Here, our open-source certificate software recomputes every hash from the files and

checks the archive's signatures, the timestamps, and the ledger, validating record consistency. The record check runs no analysis, typically takes about a minute, needs no outside service when run offline, and, as long as the cryptography it uses stays infeasible to break, works the same way decades from now.

Finally, a *re-execution check* is designed for a future researcher who wishes to build on the research or to test whether the record was faked. Conveniently, our protocol enables a researcher to rerun any single step, without running those that came before or after, whenever all of that step's starting files are stored; the certificate reports which steps qualify. The researcher can also rerun the entire analysis and see whether their results agree with the certified ones within the determinism tier. A rerun takes as long as the original, uses the deposited libraries or image, and is judged only when the researcher's software environment matches the recorded one; a rerun that agrees gives the researcher a tested starting point for new work.

Each Check Catches Frauds That Evade the One Before. A certificate is evidence of only record consistency and replication. By all accounts failure to replicate is far more often due to honest error than fraud, but we aim for certification to hold up even against an author acting in bad faith (in security terms, a "threat actor"). We also consider cases where an author and archive collude to produce a fraudulent result (which is presumably even rarer because we know of no such case). We now summarize how each type of certification check catches frauds that pass the one before it, and the permanent public record reaches what passes all three. Section 4 shows what certification does not cover.

First is *passing the existence check, caught by the record check.* The existence check relies on the archive's webpage. If it is faked, the record check catches the fraud and any file changed after the certified run, such as edited output tables, swapped data files, or secretly rerun scripts.

Second is *passing the record check, caught by re-execution.* To commit fraud while passing the record check, an author must falsify the record itself: alter the analysis while it runs, such as through hardware control of the computer or a value changed in memory; edit the

record before the archive seals it; or hack the certificate software so it still reports the genuine copy's hash. A colluding archive could also sign a falsified record and add it to its ledger. In each case the stored starting files remain untouched, so a reader who reruns a falsified step with a genuine copy of the software, in an environment matching the recorded one, will obtain the honest result and discover the fraud. Each additional researcher who reruns the analysis (which is likely to happen more often with more important scholarly claims) increases the probability of detection.

Last is *passing the re-execution check, caught by the evidence trail.* Permanent public access protects science from its worst actors, whose frauds reproduce on every rerun or lie outside the run, and so pass all three checks. The certified replication dataset, its certificate, and the archive's ledger stay publicly available, and no one can backdate them. Anyone may read code that fabricates results, read the file that sets a seed missed by the dependency audit, or examine the data (like Broockman, Kalla and Aronow (2015)). Anyone can also view the number of deposits to check for seed shopping.

Levels of Assurance. Every certificate gives the same assurance on an ordinary computer; later versions of the protocol could allow or require stronger evidence if needed: records of re-executions by others, published on the archive's ledger; an attestation from protected processor hardware, for an author whose analysis runs on it; and, should it become computationally feasible, a cryptographic proof that the computation ran. A journal could then require any of these, but a certificate does not depend on them.

3.2. Why it Needs No One's Word

Under the status quo, a replication dataset is useful only if we trust the author's word that re-executing it would produce the published results. The reason re-execution alone is insufficient is that failure to replicate is ambiguous; it may imply any or all of (a) lack of closure with respect to randomness, (b) inaccurate or incomplete documentation, or (c) published results not generated by the data and methods as claimed in the article.

In contrast, an article with a certified replication dataset carries its own evidence, which

anyone can check, anywhere, in seconds, forever, without taking anyone's word for what was run or when. "Evidence" here means a record anyone can check, which no one, including the archive, can backdate, and no one can fake without risking detection. This is summarized in four features (corresponding to Panel 2 of Figure 1), each of which answers one of the four threats named in Section 3.1.2 (seed shopping, substitution, backdating, and ambiguity), even against an author acting in bad faith.

First, *replication dataset files are fixed before the certified run.* The archive hashes the deposit and computes the seed from that hash, so the seeded draws cannot precede the files; this rests on well-tested cryptography, not on trust. Seed shopping therefore needs a new deposit for each seed, and the archive's ledger publicly records every deposit under the depositor's account, so the seeds drawn from each account can be counted; certification is not anonymous.

Second, *the results are fixed after the certified run.* The certificate records the hash of every file, each step the code started from and produced, and every logged estimate, and the "witness" names the certificate's own hash. Thus, the check catches a later change to any file or the record.

Third, *the certified run is fixed in time,* meaning that a researcher trying to correct earlier mistakes will fail or leave a visible trail. The receipt and the certificate are each timestamped, so the certified run's random draws are bounded between two block times. The later bound is the time of the block that timestamps the certificate. The earlier bound is the time of the block the receipt names, since the seed is computed from that block's hash. Not even the archive can backdate either timestamp, and that earliest date is what no signature, log, or timestamp could supply on its own.

Finally, *the result is checkable by anyone.* No special permission is required, which anyone can easily recompute using only arithmetic starting from the replication dataset and the certificate. Even the seed's derivation can be verified. The archive's word is taken only for the facts only it holds (who deposited and whether a revised deposit and an earlier deposit it names as its predecessor have the same uploader), and for not using its seed key to help an author. That the run happened as recorded rests on the open-source certificate software,

which any reader can test by re-running recorded steps.

3.3. How Everyone Benefits

Certification lowers the cost of verification by changing who runs the analysis and how often: the author runs it once more, on their machine that already holds their software environment, and everyone afterward checks the record instead of rebuilding and rerunning it from scratch. Panel 3 of Figure 1 shows the gains party by party. Because several gains reach more than one party, we take them here by the problem in Section 2 that each one solves, with what certification costs in the same terms.

First, *policy failures*. Current replication datasets, even when required by journal policies, carry no evidence that their code runs or reproduces the article's results, and journal verification catches only some issues. In contrast, a certified replication dataset carries evidence that its analysis ran on exactly its files and produced exactly the results it records, which the copyeditor or author checks during copy editing to connect the certificate to the publication. Because checking time does not grow with the length of the computation, the evidence extends to analyses too long for a data editor to re-execute, which happen because published articles often represent the computational cutting edge. The closure runs, of the kind authors already make before submission anyway, now leave a record: their report of how closely the two runs agreed, with the estimates each produced, is deposited with the replication dataset and fixed before the archive's seed exists.

Second, *time*. Data editors presently must rebuild the author's software environment, rerun their analyses, and ask the author for whatever is missing. This takes months per round before acceptance, each later researcher rebuilds it all from scratch, and authors answer questions about their deposits for years. Certification replaces publication delays and later questions with one more run by the author after conditional acceptance, on their machine that already holds the hardware, software, libraries, and data, so that the run comes down to nearly one command. A future reader or researcher attempting a replication confirms that the record is intact with the record check, which is fast and does not require recreating the author's software environment. The record check answers the questions that

would otherwise go to the author. Confirming that a step still produces its recorded outputs takes that step's own running time. The author pays by making one extra run.

Third, *money*. Journals today pay for the data editors, staff, and computing that re-execution requires. For a certified paper, journal staff only need an existence check (webpage lookup) at final acceptance with no technical choices or special expertise required. Nor does a journal need a license for proprietary software such as Stata, since the author's own run is the one certified, and readers see its logged estimates without one. Archives pay to operate the software that performs the deposit and seal stages, and for storage of the record, which includes the files the run produces. The archive's institutional work of custody, versions, backups, metadata, terms of use, restricted access, and indefinite storage continues, as before.

Finally, *rancor*. Today a failed replication becomes a dispute about what the author did, because no shared record says what was run. Certification gives the scientific community a public record of precisely what was run for every certified paper, so a dispute over a failed replication turns on what the record shows, which both sides can check, instead of on what the author says was run. The publication and its certified replication dataset can now stand independently of the person who created them. Disputes will continue, but primarily about future research, as should be the case.

3.4. Extensions to Confidential Data

We now extend our certification protocol to confidential data, requiring access levels that are not fully public, such as requiring click-through licenses, guestbook signatures, signed contracts, committee approvals, air gapped facilities, and more. We distinguish three types by who is permitted to hold the files: the archive, only a secure facility, or only the author under a license. We assume the archive may publish the data's metadata in any of these categories (otherwise no one would know the data existed in the first place). The guarantees weaken as the files move further from the archive's custody. See Section S3.5 for details.

First, *restricted data the archive is allowed to hold* require only trivial changes to the protocol. The certificate includes no data values, and our summaries of the data are hashes of the data mixed with a secret random value, so that no one can confirm a guess about

a file's contents; cryptographers call these *hiding commitments*. Before release, the author verifies that the certificate does not include private information. A later reader must then satisfy the archive's access rules for this dataset before checking the files.

Second, *data only accessible inside a secure facility*, such as a federal research data center, is not reachable by the archive, but its notary work needs only the hashes, so the facility takes over the work that uses the files. The author makes the closure runs and the certified run inside the facility, and only hiding commitments, the seed, and a certificate whose estimates pass the facility's review for disclosure leave it. The archive computes the seed from the hiding commitments; facility staff run the record check on the files and sign the result, which the archive verifies before sealing; and the facility keeps the exact files and software environment for a stated period. The certificate is then made public. Anyone can check the signatures, dates, ledger, and deposited code, and a reader admitted to the facility following the same rules as the author did can check the data or rerun the analysis. The cost here is trust in the facility's signed check and how long they keep the data, in addition to cumbersome rules that prohibit installing outside software, accessing familiar materials, or even seeing data or intermediate results.

Finally, *data licensed under terms that forbid archival deposit*, such as from commercial vendors, are analyzed on the author's computer as usual and enter the deposit only as hiding commitments and metadata naming the source and version (such as restricted data used in recent papers in the American Economic Association's journals; Vilhuber 2024). Everything but the data is certified in full, and the seed still follows the data's hashes, but only the author checks the data before sealing. A later reader can check them only by obtaining the same license and version of the data, which vendor is trusted to maintain; the archive then supplies the file of secret values needed to check the commitments.

4. What Certification Does Not Cover

What lies outside the certificate's evidence of record consistency and replication forms four groups, which we describe as follows. For details, see Section S5.

First, *a certificate is evidence about a computation, not about the science*: not that the

analysis answers the paper's question, avoids p-hacking, is free of model dependence, holds in fresh data, or contains genuine data. It locks down one run from the deposited files forward, made after conditional acceptance. It dates the record and the random draws, not the substantive work; binds output files and logged estimates, not the printed numbers which is left to the author and journal copyeditor; and shows which account deposited the files in the archive, not who wrote them.

Second, *practical limits bound which analyses can be certified.* The protocol represents a data file larger than the archive is willing to store only with its hash and where it can be obtained, and so re-executing a step that reads it requires finding a copy with that hash. A folder of millions of separate small files, such as one per social media post, would need millions of hashes, so the protocol hashes it as a single unit. Outputs produced by an API, such as a large language model, that change over time can be dealt with in several ways, none completely satisfactory: (a) if the old version of the API remains available, such as for open source models, the protocol could allow the certified run to make calls to the API, although in this case a future researcher may need to host the then outdated API; (b) the protocol could recompute only a random sample of the items, chosen with the archive's seed after the file was fixed; or (c) intermediate output files from the API can be saved, and the replication dataset certified from that point forward. We also note that some complicated workflows involving multiple computing environments may require modifications of version 1 of the protocol for certification.

Third, *how evidence ages.* The record check works as long as the archive's files survive and the archive re-timestamps them in time if Bitcoin is ever abandoned; re-execution works only while the recorded software can be run, after which the software would need to be rewritten. The cryptography guarantees that the record is intact and dated, not that it stays available: data availability rests on the archive's custody and backups, as it does today. Deposited add-on libraries, or a container image, hold the installed software, not a list of libraries that may later disappear, but an image runs only on the kind of processor it was built for, such as Intel's or Apple's, or in emulation software. Software whose license forbids deposit cannot be included and so existence and record checks are straightforward, but re-execution requires

first obtaining a license to the same software version.

Finally, *what a bad actor can still do.* Frauds that pass all three checks remain on the permanent public record at risk of being found. This includes results fabricated in the code; code that reads a file or sets a seed the dependency audit misses; and seed shopping paid for with a new, recorded deposit per seed. Re-execution catches a falsified step only when a future researcher reruns it. In addition, an archive colluding with an author, or anyone who steals the private half of the archive's seed key, can compute in advance the seed a planned deposit would receive, which allows seed shopping that leaves no record; and a stolen signing key alone yields certificates the ledger lacks, which the record check reveals against a later copy of the ledger the archive cannot revise.

5. Prior Work

We build on three bodies of prior work: (1) designs that give evidence about a published analysis; (2) the tools the protocol borrows; and (3) methods that date work by drawing public randomness after a commitment. (Section S6 summarizes our contributions.)

First, the designs, in order of increasing assurance. File fingerprints show whether a file changed (The Dataverse Project, 2026), as do the file hashes that RO-Crate and Croissant, formats that bundle research files with machine-readable descriptions, can carry (Soiland-Reyes et al., 2022; Akhtar et al., 2024); verifiable computational results store each published result under an identifier the article cites (Gavish and Donoho, 2011). Tools that record a run on the author's machine, by tracing the files it opens so that others can repeat it (ReproZip, SciUnit) or by logging the steps of a Python script (noWorkflow), and hosted platforms that run the analysis in a stored computing environment (Whole Tale, Code Ocean, Renku, Binder) rest on the word of whoever recorded or hosted the run, until someone re-executes it (Chirigati et al., 2016; Ton That et al., 2017; Murta et al., 2015; Brinckman et al., 2019; Clyburne-Sherin, Fei and Green, 2019; Roškar et al., 2023; Project Jupyter et al., 2018; Trisovic et al., 2020). Signed provenance, graded by the framework SLSA (Supply-chain Levels for Software Artifacts), and signed execution records that authors can neither alter nor disown add a signature but still rest on the word of whoever signed (Open Source Security

Foundation, 2026; Keita and Homan, 2026). TRACE, whose goal is similar to ours, requires its own platform to run and attest to the workflow, but this lets it reach data usable only inside secure facilities (Vilhuber et al., 2023; Li et al., 2025). Journal data editors, the named checkers of CODECHECK, and the French certification agency cascad re-run the analysis themselves, at the cost of a qualified person per paper (Nüst and Eglen, 2021; Pérignon et al., 2019). Trusted hardware signs output produced inside a protected processor, on the word of the chip's maker (Costan and Devadas, 2016; Li, 2026); cryptographic proofs of computation need no such trust but remain far too costly today for general data analysis (Goldwasser, Micali and Rackoff, 1989; Walfish and Blumberg, 2015). A certificate needs neither a platform that vouches for the run nor a second party's re-run.

Second, the borrowed tools. The cryptography is well-tested: Merkle trees, keyed hashes, verifiable random functions (Goldberg et al., 2023), digital signatures, and public timestamps (Section S7), obtained in Bitcoin through OpenTimestamps (Todd, 2016), with a ledger modeled on the append-only logs that Certificate Transparency keeps for website certificates and Sigstore for signed software, an internet standard (Laurie, Langley and Kasper, 2013; Newman, Meyers and Torres-Arias, 2022; Birkholz et al., 2026). Blockchain timestamps of research files and pipelines exist, but show only that files existed by some time (Gipp, Meuschke and Gernandt, 2015; Wittek et al., 2021; Casas and Fidler, 2024). The recording of a run draws on running twice under frozen settings and comparing the outputs (Lamb and Zacchiroli, 2022; Bizopoulos and Bizopoulos, 2020), and on signed per-step records of inputs and outputs and the provenance workflow engines keep (Torres-Arias et al., 2019). Re-executing a random sample of recorded steps comes from distributed computing (Monrose, Wyckoff and Rubin, 1999; Du et al., 2004; Haeberlen et al., 2010; Jia et al., 2021); the deterrence random checks provide has been analyzed as well (Golle and Mironov, 2001; Barnett, Zardo and Graves, 2018).

Finally, dating work by public randomness drawn after a "commitment", a public statement that fixes the materials. Election audits (Cordero, Wagner and Dill, 2006; Lindeman and Stark, 2012), the Internet Engineering Task Force's selection of committee members (Eastlake, 2004), randomness beacons (Rabin, 1983; Kelsey et al., 2019), and a random-

ization seeded from a Bitcoin block (Saa and Stern, 2019) all draw their randomness after publishing such a commitment, and a signature can postdate a blockchain's latest block, which dates the signature rather than a run's draws (Abadi et al., 2020). Seeds hashed from the commitment itself (Fiat and Shamir, 1987) or from a preregistration (Mazor, Mazor and Mukamel, 2019), and blind analysis, which hides results under a perturbation of the data until the analysis is frozen (MacCoun and Perlmutter, 2015), leave trust to whoever commits it, a gap that running the whole study once inside protected hardware closes, at the price of ruling out re-execution (Meißner et al., 2021). A certificate needs no separately published commitment.

6. Concluding Remarks

A certificate is designed to move the scientific community closer to the idea of keeping publications based solely on the record and independent of its author. In place of the author's word, certification provides a permanent public record of exactly which data and code produced each result, dated on the public blockchain so that no one can backdate it, and verifiable by anyone in seconds. It ends publication delays and subsequent disputes over what was done, ensures that every certified result reproduces, puts errors and fraud on a record open to all, and lets anyone question a prominent author without needing their help or their word. Because science advances as researchers check, correct, and build on one another's work, making certification standard practice should rebuild credibility in science and speed the process that produced four centuries of growth in knowledge and human welfare.

References

Abadi, Aydin, Michele Ciampi, Aggelos Kiayias and Vassilis Zikas. 2020. Timed Signatures and Zero-Knowledge Proofs—Timestamping in the Blockchain Era. In *Applied Cryptography and Network Security (ACNS 2020)*. Lecture Notes in Computer Science Springer pp. 335–354.

Akhtar, Mubashara, Omar Benjelloun, Costanza Conforti, Luca Foschini, Joan Giner-Miguelez, Pieter Gijsbers, Sujata Goswami, Nitisha Jain, Michalis Karamousadakis, Michael Kuchnik, Satyapriya Krishna, Sylvain Lesage, Quentin Lhoest, Pierre Marcenac, Manil Maskey, Peter Mattson, Luis Oala, Hamidah Oderinwale, Pierre Ruyssen, Tim Santos, Rajat Shinde, Elena Simperl, Arjun Suresh, Goeffry Thomas, Slava Tykhonov,

Joaquin Vanschoren, Susheel Varma, Jos van der Velde, Steffen Vogler, Carole-Jean Wu and Luyao Zhang. 2024. Croissant: A Metadata Format for ML-Ready Datasets. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Datasets and Benchmarks Track*. pp. 82133–82148. https://doi.org/10.52202/079017-2610

Barnett, Adrian G., Pauline Zardo and Nicholas Graves. 2018. "Randomly Auditing Research Labs Could Be an Affordable Way to Improve Research Quality: A Simulation Study." *PLOS ONE* 13(4):e0195613.

Birkholz, Henk, Antoine Delignat-Lavaud, Cédric Fournet, Yogesh Deshpande and Steve Lasker. 2026. "An Architecture for Trustworthy and Transparent Digital Supply Chains." RFC 9943, RFC Editor. https://www.rfc-editor.org/rfc/rfc9943

Bizopoulos, Paschalis and Dimitris Bizopoulos. 2020. "Reconciler: A Workflow for Certifying Computational Research Reproducibility." arXiv:2005.12660v3.

Brinckman, Adam, Kyle Chard, Niall Gaffney, Mihael Hategan, Matthew B. Jones, Kacper Kowalik, Sivakumar Kulasekaran, Bertram Ludäscher, Bryce D. Mecum, Jarek Nabrzyski, Victoria Stodden, Ian J. Taylor, Matthew J. Turk and Kandace Turner. 2019. "Computing Environments for Reproducibility: Capturing the "Whole Tale"." *Future Generation Computer Systems* 94:854–867. https://doi.org/10.1016/j.future.2017.12.029

Brodeur, Abel, Derek Mikola, Nikolai Cook et al. 2024. Mass Reproducibility and Replicability: A New Hope. Discussion Paper 16912 IZA Institute of Labor Economics. https://docs.iza.org/dp16912.pdf

Brodeur, Abel, Derek Mikola, Nikolai Cook et al. 2026. "Reproducibility and Robustness of Economics and Political Science Research." *Nature* 652:151–156. https://doi.org/10.1038/s41586-026-10251-x

Broockman, David, Joshua Kalla and Peter Aronow. 2015. "Irregularities in LaCour (2014)." Stanford University. https://stanford.edu/~dbroock/broockman_kalla_aronow_lg_irregularities.pdf

Casas, Santiago and Christian Fidler. 2024. "TOPO: Time-Ordered Provable Outputs." arXiv:2411.00072 [astro-ph.IM]. https://arxiv.org/pdf/2411.00072

Chang, Andrew C. and Phillip Li. 2022. "Is Economics Research Replicable? Sixty Published Papers from Thirteen Journals Say 'Often Not'." *Critical Finance Review* 11(1):185–206. https://doi.org/10.1561/104.00000053

Chirigati, Fernando, Rémi Rampin, Dennis Shasha and Juliana Freire. 2016. ReproZip: Computational Reproducibility With Ease. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD)*. pp. 2085–2088. https://doi.org/10.1145/2882903.2899401

Christensen, Garret and Edward Miguel. 2018. "Transparency, Reproducibility, and the Credibility of Economics Research." *Journal of Economic Literature* 56(3):920–980. https://doi.org/10.1257/jel.20171350

Clyburne-Sherin, April, Xu Fei and Seth Ariel Green. 2019. "Computational Reproducibility

via Containers in Psychology." *Meta-Psychology* 3. https://doi.org/10.15626/MP.2018.892

Cordero, Arel, David Wagner and David Dill. 2006. The Role of Dice in Election Audits—Extended Abstract. In *IAVoSS Workshop on Trustworthy Elections (WOTE 2006)*. https://people.eecs.berkeley.edu/~daw/papers/dice-wote06.pdf

Costan, Victor and Srinivas Devadas. 2016. "Intel SGX Explained." IACR Cryptology ePrint Archive, Report 2016/086. https://eprint.iacr.org/2016/086

Data Access and Research Transparency (DA-RT). 2015. "Data Access and Research Transparency (DA-RT): A Joint Statement by Political Science Journal Editors." *State Politics & Policy Quarterly* 15(4):423–424. https://doi.org/10.1177/1532440015598318

Dewald, William G., Jerry G. Thursby and Richard G. Anderson. 1986. "Replication in Empirical Economics: The Journal of Money, Credit and Banking Project." *American Economic Review* 76(4):587–603. https://www.jstor.org/stable/1806061

Du, Wenliang, Jing Jia, Manish Mangal and Mummoorthy Murugesan. 2004. Uncheatable Grid Computing. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004)*. pp. 4–11.

Eastlake, 3rd, Donald E. 2004. "Publicly Verifiable Nominations Committee (NomCom) Random Selection." RFC 3797, Internet Engineering Task Force.

Eubank, Nicholas. 2016. "Lessons from a Decade of Replications at the Quarterly Journal of Political Science." *PS: Political Science & Politics* 49(2):273–276. https://www.nickeubank.com/wp-content/uploads/2019/10/Eubank_LessonsFromADecadeOfReplications.pdf

Ferguson, Joel, Rebecca Littman, Garret Christensen, Elizabeth Levy Paluck, Nicholas Swanson, Zenan Wang, Edward Miguel, David Birke and John-Henry Pezzuto. 2023. "Survey of Open Science Practices and Attitudes in the Social Sciences." *Nature Communications* 14:5401. https://doi.org/10.1038/s41467-023-41111-1

Fiat, Amos and Adi Shamir. 1987. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology—CRYPTO '86*. Vol. 263 of *Lecture Notes in Computer Science* Springer pp. 186–194.

Fišar, Miloš, Ben Greiner, Christoph Huber, Elena Katok, Ali I. Ožkeš and the Management Science Reproducibility Collaboration. 2024. "Reproducibility in Management Science." *Management Science* 70(3):1343–1356. https://doi.org/10.1287/mnsc.2023.03556

Gavish, Matan and David L. Donoho. 2011. "A Universal Identifier for Computational Results." *Procedia Computer Science* 4:637–647.

Gipp, Bela, Norman Meuschke and André Gernandt. 2015. Decentralized Trusted Timestamping using the Crypto Currency Bitcoin. In *Proceedings of the iConference 2015*. Newport Beach, CA: . https://arxiv.org/pdf/1502.04015

Goldberg, Sharon, Leonid Reyzin, Dimitrios Papadopoulos and Jan Včelák. 2023. "Verifiable Random Functions (VRFs)." RFC 9381, RFC Editor. https://www.rfc-editor.org/rfc/rfc9381

Goldwasser, Shafi, Silvio Micali and Charles Rackoff. 1989. "The Knowledge Complexity of Interactive Proof Systems." *SIAM Journal on Computing* 18(1):186–208. https://doi.org/10.1137/0218012

Golle, Philippe and Ilya Mironov. 2001. Uncheatable Distributed Computations. In *Topics in Cryptology — CT-RSA 2001*. Springer pp. 425–440. https://doi.org/10.1007/3-540-45353-9_31

Haeberlen, Andreas, Paarijaat Aditya, Rodrigo Rodrigues and Peter Druschel. 2010. Accountable Virtual Machines. In *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI '10)*. pp. 119–134.

Hardwicke, Tom E., Maya B. Mathur, Kyle MacDonald, Gustav Nilsonne, George C. Banks, Mallory C. Kidwell, Alicia Hofelich Mohr, Elizabeth Clayton, Erica J. Yoon, Michael Henry Tessler, Richie L. Lenne, Sara Altman, Bria Long and Michael C. Frank. 2018. "Data Availability, Reusability, and Analytic Reproducibility: Evaluating the Impact of a Mandatory Open Data Policy at the Journal Cognition." *Royal Society Open Science* 5(8):180448. https://doi.org/10.1098/rsos.180448

Herndon, Thomas, Michael Ash and Robert Pollin. 2014. "Does High Public Debt Consistently Stifle Economic Growth? A Critique of Reinhart and Rogoff." *Cambridge Journal of Economics* 38(2):257–279. https://doi.org/10.1093/cje/bet075

INFORMS. 2025. "Submission Fee FAQ." Management Science. https://pubsonline.informs.org/page/mnsc/submission-fee-faq

Ioannidis, John P. A., David B. Allison, Catherine A. Ball, Issa Coulibaly, Xiangqin Cui, Aedín C. Culhane, Mario Falchi, Cesare Furlanello, Laurence Game, Giuseppe Jurman, Jon Mangion, Tapan Mehta, Michael Nitzberg, Grier P. Page, Enrico Petretto and Vera van Noort. 2009. "Repeatability of published microarray gene expression analyses." *Nature Genetics* 41(2):149–155. https://doi.org/10.1038/ng.295

Jacoby, William G. 2018. "The American Journal of Political Science Replication and Verification Policy." American Journal of Political Science. https://ajps.org/wp-content/uploads/2019/03/ajps-replic-and-verif-policy-2-27-18.pdf

Jacoby, William G., Sophia Lafferty-Hess and Thu-Mai Christian. 2017. "Should Journals Be Responsible for Reproducibility?" Inside Higher Ed. https://www.insidehighered.com/blogs/rethinking-research/should-journals-be-responsible-reproducibility

Jia, Hengrui, Mohammad Yaghini, Christopher A. Choquette-Choo, Natalie Dullerud, Anvith Thudi, Varun Chandrasekaran and Nicolas Papernot. 2021. Proof-of-Learning: Definitions and Practice. In *2021 IEEE Symposium on Security and Privacy (S&P)*. pp. 1039–1056. https://arxiv.org/abs/2103.05633

Kahneman, Daniel. 2014. "A New Etiquette for Replication." *Social Psychology* 45(4):310–311. https://doi.org/10.1027/1864-9335/a000202

Keita, Mamadou K. and Christopher Homan. 2026. "Computer Science Conferences Should Require Nonrepudiable Experimental Results." arXiv:2605.08586.

Kelsey, John, Luís T. A. N. Brandão, Rene Peralta and Harold Booth. 2019. A Reference for Randomness Beacons: Format and Protocol Version 2. Technical Report NIST IR 8213 (Initial Public Draft) National Institute of Standards and Technology. https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8213-draft.pdf

King, Gary. 1995. "Replication, Replication." *PS: Political Science & Politics* 28(3):444–452. https://gking.harvard.edu/files/replication.pdf

King, Gary. 2007. "An Introduction to the Dataverse Network as an Infrastructure for Data Sharing." *Sociological Methods & Research* 36(2):173–199. https://gking.harvard.edu/files/dvn.pdf

Lamb, Chris and Stefano Zacchiroli. 2022. "Reproducible Builds: Increasing the Integrity of Software Supply Chains." *IEEE Software* 39(2):62–70. https://arxiv.org/pdf/2104.06020

Laurie, Ben, Adam Langley and Emilia Kasper. 2013. "Certificate Transparency." RFC 6962, RFC Editor. https://www.rfc-editor.org/rfc/rfc6962.txt

Li, Jiasun. 2026. "Trusted-Execution Environment (TEE) for Solving the Replication Crisis in Academia." arXiv:2603.24878.

Li, Meng, Timothy McPhillips, Craig Willis, Nikolaus Parulian, Bertram Ludäscher, Kacper Kowalik, Lars Vilhuber, Thu-Mai Lewis and Mandy Gooch. 2025. "TROV—A Model and Vocabulary for Describing Transparent Research Objects." *International Journal of Digital Curation* 19(1):7.

Lindeman, Mark and Philip B. Stark. 2012. "A Gentle Introduction to Risk-Limiting Audits." *IEEE Security & Privacy* 10(5):42–49.

Lupia, Arthur and Colin Elman. 2014. "Openness in Political Science: Data Access and Research Transparency." *PS: Political Science & Politics* 47(1):19–42. https://doi.org/10.1017/S1049096513001716

Luttmer, Erzo F. P. 2026. "Report of the Editor, American Economic Review." *AEA Papers and Proceedings* 116:742–760. https://doi.org/10.1257/pandp.116.742

MacCoun, Robert and Saul Perlmutter. 2015. "Blind Analysis: Hide Results to Seek the Truth." *Nature* 526:187–189. https://doi.org/10.1038/526187a

Mazor, Matan, Noam Mazor and Roy Mukamel. 2019. "A Novel Tool for Time-Locking Study Plans to Results." *European Journal of Neuroscience* 49(9):1149–1156. https://matanmazor.github.io/files/papers/mazor2019novel.pdf

Meißner, Echo, Felix Engelmann, Frank Kargl and Benjamin Erb. 2021. PeQES: A Platform for Privacy-Enhanced Quantitative Empirical Studies. In *Proceedings of the 36th ACM/SIGAPP Symposium on Applied Computing (SAC '21)*. pp. 1226–1234. https://arxiv.org/pdf/2103.05544

Miske, Olivia et al. 2026. "Investigating the reproducibility of the social and behavioural sciences." *Nature* 652(8108):126–134. https://doi.org/10.1038/s41586-026-10203-5

Monrose, Fabian, Peter Wyckoff and Aviel D. Rubin. 1999. Distributed Execution with

Remote Audit. In *Proceedings of the Network and Distributed System Security Symposium (NDSS '99)*. https://www.ndss-symposium.org/wp-content/uploads/2017/09/Distributed-Execution-with-Remote-Audit.pdf

Murta, Leonardo, Vanessa Braganholo, Fernando Chirigati, David Koop and Juliana Freire. 2015. noWorkflow: Capturing and Analyzing Provenance of Scripts. In *Provenance and Annotation of Data and Processes: 5th International Provenance and Annotation Workshop (IPAW 2014)*. Vol. 8628 of *Lecture Notes in Computer Science* Springer pp. 71–83. https://doi.org/10.1007/978-3-319-16462-5_6

National Academies of Sciences, Engineering, and Medicine. 2019. *Reproducibility and Replicability in Science*. Washington, DC: The National Academies Press. https://doi.org/10.17226/25303

Newman, Zachary, John Speed Meyers and Santiago Torres-Arias. 2022. Sigstore: Software Signing for Everybody. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. pp. 2353–2367. https://doi.org/10.1145/3548606.3560596

Nosek, Brian A., George Alter, George C. Banks et al. 2015. "Promoting an Open Research Culture." *Science* 348(6242):1422–1425. https://doi.org/10.1126/science.aab2374

Nüst, Daniel and Stephen J. Eglen. 2021. "CODECHECK: an Open Science initiative for the independent execution of computations underlying research articles during peer review to improve reproducibility." *F1000Research* 10:253. Version 2. https://f1000research.com/articles/10-253/v2/pdf

Open Source Security Foundation. 2026. "SLSA: Supply-chain Levels for Software Artifacts." https://slsa.dev/.

Peng, Roger D. 2011. "Reproducible Research in Computational Science." *Science* 334(6060):1226–1227. https://doi.org/10.1126/science.1213847

Pérignon, Christophe, Kamel Gadouche, Christophe Hurlin, Roxane Silberman and Eric Debonnel. 2019. "Certify reproducibility with confidential data." *Science* 365(6449):127–128. https://doi.org/10.1126/science.aaw2825

Pimentel, João Felipe, Leonardo Murta, Vanessa Braganholo and Juliana Freire. 2019. A Large-Scale Study About Quality and Reproducibility of Jupyter Notebooks. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. pp. 507–517. https://doi.org/10.1109/MSR.2019.00077

Project Jupyter, Matthias Bussonnier, Jessica Forde, Jeremy Freeman, Brian Granger, Tim Head, Chris Holdgraf, Kyle Kelley, Gladys Nalvarte, Andrew Osheroff, M Pacer, Yuvi Panda, Fernando Perez, Benjamin Ragan-Kelley and Carol Willing. 2018. Binder 2.0 – Reproducible, Interactive, Sharable Environments for Science at Scale. In *Proceedings of the 17th Python in Science Conference (SciPy 2018)*. pp. 113–120. https://doi.org/10.25080/Majora-4af1f417-011

Rabin, Michael O. 1983. "Transaction Protection by Beacons." *Journal of Computer and*

System Sciences 27(2):256–267. https://doi.org/10.1016/0022-0000(83)90042-9

Rainey, Carlisle, Harley Roe, Qing Wang and Hao Zhou. 2025. "Data and Code Availability in Political Science Publications from 1995 to 2022." *PS: Political Science & Politics* 58(2):339–345. https://doi.org/10.1017/S1049096524001276

Reiss, Michael V., Susanne J. Adler, Christopher Barrie et al. 2026. "Improving Computational Reproducibility in the Social Sciences." *Nature Human Behaviour*. https://doi.org/10.1038/s41562-026-02570-w

Roškar, Rok, Chandrasekhar Ramakrishnan, Michele Volpi, Fernando Perez-Cruz, Lilian Gasser, Firat Ozdemir, Patrick Paitz, Mohammad Alisafaee, Philipp Fischer, Ralf Grubenmann, Eliza Harris, Tasko Olevski, Carl Remlinger, Luis Salamanca, Elisabet Capon Garcia, Lorenzo Cavazzi, Jakub Chrobasik, Darlin Cordoba Osnas, Alessandro Degano, Jimena Dupre, Wesley Johnson, Eike Kettner, Laura Kinkead, Sean D. Murphy, Flora Thiebaut and Olivier Verscheure. 2023. Renku: A Platform for Sustainable Data Science. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023), Datasets and Benchmarks Track*. pp. 42161–42173. https://doi.org/10.52202/075280-1827

Saa, Olivia and Julio Michael Stern. 2019. Auditable Blockchain Randomization Tool. In *Proceedings of the 39th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering (MaxEnt 2019)*. Vol. 33 of *Proceedings* MDPI p. 17.

Samuel, Sheeba and Daniel Mietchen. 2024. "Computational reproducibility of Jupyter notebooks from biomedical publications." *GigaScience* 13:giad113. https://doi.org/10.1093/gigascience/giad113

Schwartz, Matthew, Isaiah Andrews and Jesse M. Shapiro. 2026. An LLM Workflow That Reproduces, Improves, and Extends Published Economics Research. Working Paper 35782 National Bureau of Economic Research. https://www.nber.org/papers/w35782

Soiland-Reyes, Stian, Peter Sefton, Mercè Crosas, Leyla Jael Castro, Frederik Coppens, José M. Fernández, Daniel Garijo, Björn Grüning, Marco La Rosa, Simone Leo, Eoghan Ó Carragáin, Marc Portier, Ana Trisovic, RO-Crate Community, Paul Groth and Carole Goble. 2022. "Packaging Research Artefacts with RO-Crate." *Data Science* 5(2):97–138. https://doi.org/10.3233/DS-210053

Stodden, Victoria, Jennifer Seiler and Zhaokun Ma. 2018. "An Empirical Analysis of Journal Policy Effectiveness for Computational Reproducibility." *Proceedings of the National Academy of Sciences* 115(11):2584–2589. https://doi.org/10.1073/pnas.1708290115

The Dataverse Project. 2026. "Universal Numerical Fingerprint (UNF)." https://guides.dataverse.org/en/latest/developers/unf/index.html.

Todd, Peter. 2016. "OpenTimestamps: Scalable, Trust-Minimized, Distributed Timestamping with Bitcoin." https://petertodd.org/2016/opentimestamps-announcement.

Ton That, Dai Hai, Gabriel Fils, Zhihao Yuan and Tanu Malik. 2017. Sciunits: Reusable Research Objects. In *2017 IEEE 13th International Conference on e-Science (e-Science)*.

pp. 374–383. https://doi.org/10.1109/eScience.2017.51

Torres-Arias, Santiago, Hammad Afzali, Trishank Karthik Kuppusamy, Reza Curtmola and Justin Cappos. 2019. in-toto: Providing Farm-to-Table Guarantees for Bits and Bytes. In *28th USENIX Security Symposium*. USENIX Association pp. 1393–1410. https://www.usenix.org/system/files/sec19-torres-arias.pdf

Trisovic, Ana, Matthew K. Lau, Thomas Pasquier and Mercè Crosas. 2022. "A Large-Scale Study on Research Code Quality and Execution." *Scientific Data* 9:60. https://www.nature.com/articles/s41597-022-01143-6.pdf

Trisovic, Ana, Philip Durbin, Tania Schlatter, Gustavo Durand, Sonia Barbosa, Danny Brooke and Mercè Crosas. 2020. Advancing Computational Reproducibility in the Dataverse Data Repository Platform. In *Proceedings of the 3rd International Workshop on Practical Reproducible Evaluation of Computer Systems (P-RECS '20)*. ACM pp. 15–20. https://doi.org/10.1145/3391800.3398173

Vilhuber, Lars. 2020. "Reproducibility and Replicability in Economics." *Harvard Data Science Review* 2(4). https://hdsr.mitpress.mit.edu/pub/fgpmpj1l/download/pdf

Vilhuber, Lars. 2024. "Report of the AEA Data Editor." *AEA Papers and Proceedings* 114:878–890. https://doi.org/10.1257/pandp.114.878

Vilhuber, Lars. 2025. "Impacts of the AEA Verification Process." Office of the AEA Data Editor. https://aeadataeditor.github.io/aea-cumulative-summary/impacts_of_aea_data_editing.html

Vilhuber, Lars, Hyuk Harry Son, Meredith Welch, David N. Wasser and Michael Darisse. 2022. "Teaching for Large-Scale Reproducibility Verification." *Journal of Statistics and Data Science Education* 30(3):274–281. https://doi.org/10.1080/26939169.2022.2074582

Vilhuber, Lars, Thu-Mai Christian, Michał Kowalik, Bertram Ludäscher, Timothy McPhillips and Craig Willis. 2023. "TRACE: TRAnsparency CErtified — Trusting Computational Research Without Repeating It." Zenodo. https://doi.org/10.5281/zenodo.8000023

Walfish, Michael and Andrew J. Blumberg. 2015. "Verifying Computations Without Reexecuting Them." *Communications of the ACM* 58(2):74–84. https://dl.acm.org/doi/pdf/10.1145/2641562

Wittek, Kevin, Neslihan Wittek, James Lawton, Iryna Dohndorf, Alexander Weinert and Andrei Ionita. 2021. A Blockchain-Based Approach to Provenance and Reproducibility in Research Workflows. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. pp. 1–6.

Xu, Yiqing and Leo Yang Yang. 2026. "Scaling Reproducibility: An AI-Assisted Workflow for Large-Scale Replication and Reanalysis." arXiv:2602.16733. https://arxiv.org/pdf/2602.16733