

Source-Recognized Citation Watermarks

A Protocol for Bilateral and Reusable Source Attestation in Natural-Language Text

Protocol 0.1

Status: protocol specification, version 0.1.

Abstract

Visible citations are fragile. A hyperlink or footnote disappears as soon as the text is pasted into an email, scraped into a corpus, or dropped into a language-model prompt. This specification defines Source-Recognized Citation Watermarks (SRCW): a way to carry a citation inside the prose itself, with the cited source deciding what the mark means.

The protocol is simple. A cited source publishes an attestation selector, a rule for picking characters out of a text, in its llms.txt file. A publisher that cites one of the source’s documents derives a short payload from that document’s text and writes its piece so that the selector picks out the payload. The source retrieves the piece, applies the selector, and confirms the mark. It lists every page it has verified, whether the publisher registered the page or the source found it later as a copy, under the payload it recognizes.

SRCW is not a new watermarking technique. Null ciphers, linguistic steganography, LLM watermarks, and provenance systems already exist. What we add is the division of roles and a way to publish approvals on the web. We define the protocol, the selector notation, the payload rule, and the llms.txt convention. We also set out what a verified record tells a reader and what it does not. SRCW is meant to sit alongside visible citations, DOIs, and signed metadata, not replace them.

1. Introduction

A visible citation tells a reader where a statement came from. It works well until the text moves. Copy a paragraph into an email, paste it into a chat window, or scrape it into a training set, and the hyperlink, the footnote, and the HTML metadata are usually gone. The words survive. The attribution does not.

Text watermarking solves a different problem. It puts a detectable signal in the words themselves, through letter positions, word choice, syntax, or the randomness a language model uses to pick its next token. Early natural-language watermarking aimed at copyright and tamper detection [1]-[5]. Recent LLM watermarking shows that a detector holding the right key can estimate whether a passage was generated with the watermark switched on [6]-[12]. Anthropic’s Claude watermark, for example, is a variant of Google DeepMind’s SynthID-Text: token choice uses keyed randomness, detection is probabilistic, and heavy rewriting removes the signal [12], [13].

SRCW carries the same idea into a citation between two independent sites. Throughout, exampleA.com is the cited source and exampleB.com is the publisher whose content cites or uses it. Example B controls the text. Example A controls what the watermark means. Anyone else can read Example A’s record and, because the selector and the payload rule are public, check the mark for themselves.

That is a different arrangement from a model provider’s watermark, where one organization both embeds and detects, and from a copyright watermark, where an owner marks its own asset. Section 6 covers the closest prior work. We found nothing that combines a publisher-embedded citation mark, a source-controlled meaning, and web-published approvals in this way. That is a negative search result, not a proof of novelty.

2. The Protocol

Everything follows from two published facts: Example A's selector, and a rule that turns a cited document into a payload. Given those, any publisher can mark a citation of any Example A document, and Example A can recognize the mark on any page, registered or not.

1. Example A publishes a selector. Its `llms.txt` names a selector pattern, such as 1-4-12.31-5, and the payload rule.
2. Example B derives the payload and writes the piece. It normalizes the text of the Example A document it is citing, hashes it, and turns the hash into letters, one per selector position. It writes its prose so that the selector picks out those letters.
3. Example B publishes. It may also register the page with Example A, by email or an endpoint, so that the page is verified promptly.
4. Example A verifies and lists. It retrieves the page, applies the selector, matches the letters against the payloads of its own documents, and lists the referencing URL under the matching record. Pages it finds later, including copies of Example B's text on other sites, are added to the same record.

2.1 Selector Notation

An attestation selector is a string of numbers. Periods separate sentences. Within a sentence, dashes separate character positions. So 1-4-12.31-5 means: from the first sentence, take characters 1, 4, and 12; from the second sentence, take characters 31 and 5. The payload is those five characters, in that order. Counting starts at 1 and includes spaces. Two periods together skip a sentence.

Both sides apply the selector to the same normalized form of the article text, produced by five mechanical steps:

1. Take the visible text of the article body: the words a reader would select and copy. Headings, bylines, navigation, and captions are excluded.
2. Apply Unicode Normalization Form C.
3. Replace typographic quotation marks, apostrophes, and dashes with their ASCII equivalents, and remove zero-width characters and soft hyphens.
4. Collapse every run of whitespace, including line breaks, to a single space, and trim both ends.
5. Split into sentences at every period, question mark, or exclamation mark that is followed by a space or by the end of the text. The punctuation stays with the sentence before it. As a regular expression, the split point is `(?<=[.!?])\s+`.

These rules do not try to find linguistic sentences, and they do not need to. "Dr. Smith paid 3.5 percent." splits after "Dr." and not inside "3.5". That is wrong for a grammarian and fine for a watermark, because the publisher and the source apply the same rule and get the same result. We deliberately avoid Unicode sentence segmentation [51]. Its rules are long and change between versions, and a citation mark needs agreement, not linguistics.

Steps 3 and 4 remove the most common ways a copied text differs from its original: smart quotes turned straight or straight quotes turned smart, lines re-wrapped, indentation added, double spaces removed. Numbering within sentences rather than across the whole document does the rest. An edit to sentence 3 does not move anything in sentences 1 and 2.

The payload has one letter per selector position. Five is a reasonable default. Section 2.2 says what more positions buy and what they cost.

2.2 Payload Derivation

The payload is derived from the cited document, so that everyone who cites the same document produces the same mark and nobody has to be told what the mark is.

1. Take the cited document's text and normalize it with steps 1 to 4 of Section 2.1.
2. Compute the SHA-256 digest of the UTF-8 bytes.
3. For each selector position, in order, take the next byte of the digest, reduce it modulo 26, and map it to a letter from a to z.

Comparison is case-insensitive, so a selected letter may fall at the start of a sentence. A space or punctuation mark at a selected position never matches; the writer must put a letter there.

Suppose Example A's report contains: "Participants who slept fewer than six hours showed a 12 percent drop in recall. The effect held across all age groups." Its SHA-256 digest begins 5d 94 9c f1 e2, which gives the letters p, s, a, h, s. With the selector 1-4-12.31-5, Example B needs a first sentence whose characters 1, 4, and 12 are p, s, and a, and a second sentence whose characters 31 and 5 are h and s. This works:

Press coverage of the sleep study missed its main finding. Its subjects who slept fewer than six hours recalled 12 percent fewer words.

Because the payload comes from the document rather than from an agreement, three things follow. A publisher can mark a citation without contacting Example A first. A copy of a marked paragraph stays marked, and still names the document it cites. And Example A can recognize a citation on a page it has never heard of, by comparing the extracted letters against the payloads of its own documents. What gets registered is the page, not the mark.

A payload is tied to one version of the document. If Example A edits the document, the payload for the new text differs, and Example A keeps the earlier payload listed so that existing citations still resolve. Five letters give about twelve million distinct payloads, which is ample for checking a known page against a known document. A source that scans the open web, where many pages and many sentence offsets are tested against many documents, should publish a longer selector; each extra position adds a letter.

2.3 What the Record Means

The record is the heart of the scheme. It says, in Example A's voice, "we recognize this selector and payload as a citation of this document, and these are the pages we have verified carry it." A third party who finds a paragraph in the wild can apply the selector, look up the letters in exampleA.com's llms.txt, and learn what the mark means and who else carries it.

Detection belongs to the source. This is the same position a model provider takes with its own watermark. Anthropic describes Claude's watermark as detectable to anyone who holds the key, makes detection available to organizations that have a reason to check, and describes the result as a likelihood that Claude was involved rather than proof of authorship [12]. Google frames SynthID the same way, as a confidence score to be combined with other signals [13]. SRCW follows that model. The cited source holds the selector, performs the check, and publishes the result. A verified record is the source's statement that the check passed. Section 7 sets out exactly what that statement covers. With a public pattern anyone can repeat the check; the listing is still Example A's.

3. The llms.txt Convention

llms.txt is a community proposal for a short, Markdown-formatted file that tells crawlers and language models what a site offers [46]. It is the natural place for a cited source to publish its selector and to list the pages it has verified. The proposed section for exampleA.com looks like this:

Source-recognized citation watermarks

Pages that cite this site may carry a citation watermark that this site recognizes. Use the selector pattern below with a payload derived from the cited document, as described at srcw-spec.org. Verified pages are listed here. A record means only what its fields state.

```
{ "spec": "https://srcw-spec.org/spec/0.1/spec.json",
  "pattern": "exampleA-v1",
  "attestation_selector": "1-4-12.31-5",
  "payload_rule": "sha256-letters",
  "reuse": "public",
  "register_at": "attestation@exampleA.com" }

{ "spec": "https://srcw-spec.org/spec/0.1/spec.json",
  "pattern": "exampleA-v1",
  "payload": "psahs",
  "cites": "exampleA.com/reports/result-17",
  "referencing_urls": [
    { "url": "exampleB.com/analysis/result-17", "verified": "2026-08-21" },
    { "url": "exampleD.com/weekly-roundup", "verified": "2026-08-30" },
```

```
{ "url": "exampleE.com/notes/sleep-and-memory", "verified": "2026-09-01" }
}]}
```

The first entry is the pattern: the selector, the payload rule, and where to register. The second is a record for one document. It has one payload and three referencing pages. Example B registered its page. Example D was found later and carries a copy of Example B's paragraph. Example E wrote its own piece against the same document and never registered, and was found anyway. All three are verified, and all three cite the same report.

A few conventions apply. Every entry starts with a spec field naming the version of this specification it follows, so a reader or a tool can fetch the field definitions from `srcw-spec.org` rather than guess. The one-sentence caveat at the top matters, because automated readers otherwise flatten recognized, approved, and endorsed into one claim. Each referencing entry is a URL and the date Example A verified it. The record does not say whether the publisher registered the page or Example A found it, because that makes no difference to what the entry means. The `cites` field is the canonical URL of the document the payload was derived from; where the cited item is a passage, it can carry a fragment identifier.

A site with a handful of citing pages can keep everything in `llms.txt`. A site with thousands should keep `llms.txt` as the human-readable signpost and move the records to a JSON file it links from there, for example `/.well-known/source-attestations` once that name is registered under RFC 8615 [44]. The JSON records use the same fields. Example A may sign the file if it wants to, using JCS and JWS [47], [48], but signing is not required for the base protocol.

4. Shared Patterns, Copies, and a Covert Variant

The shared pattern is the default. Example A publishes one selector, and any publisher may use it. That is the point: a new site implements the scheme by reading two things from `llms.txt`, the selector and the payload rule, and writing to them. A match tells you that the page carries a mark that follows Example A's pattern and names one of Example A's documents. It does not tell you that Example A approved the publisher. Example A's listing tells you that Example A has checked the page and found the mark.

Registration is a courtesy, not a status. A publisher that registers its page gets verified promptly. A page Example A discovers on its own, by scanning or by a tip, gets verified when it is discovered. Both end up as the same kind of entry, because both mean the same thing about the text. A copied paragraph that turns up on a third site is listed under the same payload, which is correct: the copied text still cites the document. The listing does not claim the copier obtained the text legitimately, and Example A may delist a page it does not wish to be associated with. The mark stays in the text either way.

The base protocol uses the public shared pattern. A covert extension hides both the selector and the payload. Hiding the selector alone achieves nothing, because the payload can be computed by anyone and the record names the referencing page. A source that wants a mark only it can read for a bilateral arrangement must hide the payload too: it substitutes HMAC-SHA-256 with a private key for the plain hash [29], keeps the selector unpublished, and keeps the record unlisted. Either key holder could then have made the mark, so the mark alone does not say which. This extension requires key management and a second payload rule.

Patterns can be retired. A deprecated pattern accepts no new registrations after a stated date, while pages already listed stay listed. Because payloads are tied to document versions, Example A also keeps the payload of an earlier version listed after it edits the document. Nothing in the registry can remove a mark from text that has already been copied. It only changes what the record says about it.

5. Why Put the Mark in the Words

A link is metadata. It lives beside the words, in an href, a footnote, or a meta tag, and every step that strips formatting strips it too. The mark lives in the words. It survives exactly the operations that preserve the words, and it fails under exactly the operations that change them.

That is the whole case for the mechanism, and it is also the right boundary. The operations that keep words intact are the ones where attribution is most often lost: copying a paragraph into an email or a chat, converting a page to plain text, quoting a passage in another document, chunking an article for a retrieval index, or placing it in a model's context window. In all of these the sentences arrive with their characters in order, the selector

resolves, and because the payload is derived from the cited document, the copy still names what it cites even on a page Example A has never seen. The operations that change the words, such as rewording, summarizing, and translating, produce something that is no longer the publisher’s text, and a citation of the original page would be the wrong citation anyway.

Between those two cases sits ordinary editing. A copyedit that touches one sentence breaks the positions in that sentence and no other. A selector that draws from five sentences loses one character out of five, and the verifier, which knows the payload it is looking for, sees a partial match and can report it as such. An excerpt that starts mid-article shifts the sentence numbers; the verifier can try each sentence offset in turn and report the one that matches. Both are ordinary string operations on a known target.

What remains fragile is a rewrite of a sentence the selector depends on. Example B’s authoring tool should show the selected characters and re-run the selector after every edit, so that this is caught before publication. If Example B edits the page after verification, it re-runs the selector and, if the mark no longer resolves, proposes a new one. The tool must never rewrite prose to make a mark fit; the publisher stays responsible for the words.

Other selector families could be added later, and each changes what “verified” means. Word-initial selectors, lexical and syntactic choice codes [2]-[5], [19], keyed token sampling [6]-[10], and source-grounded semantic signals [16], [20], [21] survive more kinds of change and are much harder to implement, and a statistical or semantic detector gives a score rather than an exact match. None should be standardized until its error rates have been measured. SRCW 0.1 uses character selectors only.

6. Related Work

We reviewed work in six areas: linguistic steganography, LLM watermarking, controlled-verification cryptography, commitment and time evidence, provenance and citation infrastructure, and web discovery. The references below are the closest items in each area, not an exhaustive survey, and relevant work was still appearing at the review cutoff.

Null ciphers and linguistic steganography. A null cipher hides a message in chosen characters or words of an innocent-looking text; the acrostic is the familiar example. Bennett surveys the field [1]. Atallah, Topkara, and colleagues built natural-language watermarking on lexical, syntactic, and semantic transformations [2], [3]. Chang and Clark demonstrated synonym substitution with unambiguous coding and, later, generated paraphrases [4], [5]. An attestation selector is a null cipher. The idea is old; what is new is who approves it and where the approval is published.

LLM watermarking. Kirchenbauer et al. bias token selection toward a keyed list and detect the excess [6], [7]. Christ, Gunn, and Zamir formalize undetectable watermarks [8]. Kudipudi et al. give distortion-free watermarks [9]. SynthID-Text uses tournament sampling and a learned detector at production scale [10]. Multi-bit and semantic methods carry richer payloads or resist paraphrase [19]-[21]. Impossibility results show that strong watermarking cannot be assumed against general rewriting [22]. Claude’s 2026 watermark is a SynthID-Text variant [12], [13]. These systems show controlled detection at scale, but embedding and detection stay with the model provider.

Who gets to detect. Publicly detectable watermarking lets anyone detect [11]. Multi-designated detector watermarking lets a provider name specific detectors [14]. Selective-disclosure watermarking layers the payload by audience [23]. SRCW’s shared pattern is the same idea applied to a different relationship: the detector is the cited source, and the publisher controls the text.

Source-grounded signals. In-context watermarking places instructions in a prompt so that an ordinary LLM emits detectable patterns [15]. SemTrace derives a signal from alternative factual propositions in a protected document [16]. Waterfall embeds IDs through paraphrasing [24]. SemTrace is the closest in role structure, since a document owner controls the source while another system produces the text. The difference is consent: SemTrace detects exposure, possibly covertly, whereas SRCW records a citation the publisher joins on purpose.

Provenance and web discovery. C2PA binds signed claims into manifests and can use a watermark or fingerprint as a soft binding to find a manifest after metadata is stripped [18], [35]. W3C PROV models derivation [36]. Web Annotation defines text selectors [37]. Crossref, DataCite, CiTO, Schema.org, and OpenCitations provide citation vocabularies [38]-[42]. RFC 8288, RFC 8615, and RFC 9309 govern typed links, well-known URIs, and robots.txt [43]-[45]; llms.txt is a community proposal [46]. JCS, JWS, HTTP Message Signatures, and Digest

Fields supply signing and serialization [47]-[50]. C2PA soft binding is the closest standards architecture. SRCW should interoperate with it. What SRCW adds is that the cited source, not only the creator of the text, says what the mark means.

Deferred to a later document. Designated-verifier and undeniable signatures [25]-[28], MACs [29], commitments [30], digital timestamping and transparency logs [31]-[34], zero-knowledge watermark detection [17], and tamper-tracing dual watermarks [52] are the tools one would reach for to prove intent or chronology to a hostile third party, or to detect targeted removal. That analysis is out of scope here.

7. What a Verified Record Tells You

A verified record tells you that the cited source recognizes this selector and payload as a citation of a named document, and that it has applied the selector to each listed page and found the mark. With a public pattern, it also tells you that you can repeat the check yourself.

It does not tell you anything about ownership, authorship, or legal responsibility for either text. It does not tell you that the publisher read or relied on the source, that the cited claim is true, or that the source endorses the piece. It records the source's recognition of the mark; the publisher's intent is expressed by the act of writing to the pattern, and registration adds only that the publisher said so. The listing itself you take as you would take a DOI registry's answer: as the registry's word, published under its own name.

Like any watermark, this one is a signal to combine with others, not a verdict on its own. Detector output should list what was checked rather than collapse to a single "verified" flag, and should warn when a page carries a hidden citation but no visible one. A hidden mark cannot tell a human reader what the citation covers, which is why the visible citation stays primary.

Three practical cautions apply. A copied paragraph carries the mark and, once verified, is listed like any other page; that says the copy cites the document, not that the copier had permission to copy. Two documents can in principle derive the same five letters, about one chance in twelve million per pair; a source with a large archive resolves this by publishing a longer selector. And short payloads in long texts will sometimes match by chance, especially when many sentence offsets are tried; a source that scans the open web should expect a few false positives and confirm each before listing it.

Stronger guarantees, such as a countersignature from the publisher or independent evidence of timing, are possible. They, and the situations that would call for them, are the subject of a separate document.

8. Limitations and Open Questions

Character selectors survive copying and fail under rewording of the sentences they depend on. Because the selector is public, deliberate removal is easy for anyone who wants to remove it. And the protocol asks sources to run an approval workflow, which may cost more than the benefit for many of them. Whether publishers will accept constraints on their prose, and whether sources will approve watermarks at all, are questions only deployment can answer.

The open questions we consider most important are:

1. What wording in llms.txt and detector output stops readers and agents from confusing recognition, approval, reuse, and endorsement?
2. How many selector positions does a source need before open-web scanning, with many pages, many sentence offsets, and many documents, gives acceptably few false matches?
3. Should Example A publish the payload for each document alongside it, so that publishers need not fetch and hash the text themselves, and how should edited documents be handled beyond keeping old payloads listed?
4. Should approval records eventually be expressed as C2PA assertions or W3C PROV bundles, or stay as independent JSON entries?
5. What governance should own the well-known URI, the selector notation, and the list of selector families?

9. Conclusion

SRCW is a small protocol built from old parts. A cited source publishes an attestation selector; a publisher derives the payload from the cited document and writes to it; the source verifies and lists. The mark is a null cipher whose key is the cited text itself. The record is a JSON entry in llms.txt. The contribution is to give a hidden citation a meaning that the cited party controls, and to publish that meaning where anyone can find it.

The most useful property of the design is that the mark needs no agreement to create and no registration to recognize. A new site implements it by reading one pattern, and a copy keeps naming what it cites. Detection belongs to the cited source, in the same way that detection of a model provider’s watermark belongs to the key holder, and a verified record is the source’s published statement that the check passed.

Conforming implementations use plain text, character selectors, a hash, and an llms.txt section. The protocol is intentionally fragile and testable. Public test corpora and review by provenance researchers, citation-infrastructure maintainers, and web standards engineers are encouraged.

References

- [1] K. Bennett, “Linguistic Steganography: Survey, Analysis, and Robustness Concerns for Hiding Information in Text,” CERIAS Technical Report 2004-13, Purdue University, 2004. https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/PSI000441.pdf
- [2] M. J. Atallah, V. Raskin, C. F. Hempelmann, M. Karahan, R. Sion, U. Topkara, and K. E. Triezenberg, “Natural Language Watermarking and Tamperproofing,” in Information Hiding, 2002. <https://zxr.io/research/Atallah-2002-NLW.pdf>
- [3] M. Topkara, U. Topkara, and M. J. Atallah, “Natural Language Watermarking,” SPIE, 2005. http://mercan.topkara.org/ei05_5681_45.pdf
- [4] C.-Y. Chang and S. Clark, “Practical Linguistic Steganography Using Contextual Synonym Substitution and a Novel Vertex Coding Method,” Computational Linguistics, vol. 40, no. 2, pp. 403-448, 2014. <https://aclanthology.org/J14-2006.pdf>
- [5] C.-Y. Chang and S. Clark, “Linguistic Steganography Using Automatically Generated Paraphrases,” NAACL, 2010. <https://aclanthology.org/N10-1084.pdf>
- [6] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, “A Watermark for Large Language Models,” ICML, 2023. <https://proceedings.mlr.press/v202/kirchenbauer23a.html>
- [7] J. Kirchenbauer et al., “On the Reliability of Watermarks for Large Language Models,” ICLR, 2024. <https://arxiv.org/abs/2306.04634>
- [8] M. Christ, S. Gunn, and O. Zamir, “Undetectable Watermarks for Language Models,” COLT, 2024. <https://proceedings.mlr.press/v247/christ24a.html>
- [9] R. Kuditipudi, J. Thickstun, T. Hashimoto, and P. Liang, “Robust Distortion-Free Watermarks for Language Models,” Transactions on Machine Learning Research, 2024. <https://openreview.net/forum?id=FpaCL1MO2C>
- [10] S. Dathathri et al., “Scalable Watermarking for Identifying Large Language Model Outputs,” Nature, vol. 634, pp. 818-823, 2024. <https://www.nature.com/articles/s41586-024-08025-4>
- [11] J. Fairoze, S. Garg, S. Jha, S. Mahlouljifar, M. Mahmood, and M. Wang, “Publicly-Detectable Watermarking for Language Models,” IACR ePrint 2023/1661, revised 2025. <https://eprint.iacr.org/2023/1661>
- [12] Anthropic, “How Claude’s Text Watermark Works,” 14 August 2026, updated 1 September 2026. <https://www.anthropic.com/news/claude-text-watermark>
- [13] Google DeepMind, “SynthID: Tools for Watermarking and Detecting LLM-Generated Text,” official documentation. <https://ai.google.dev/responsible/docs/safeguards/synthid>
- [14] Z. Huang, G. Zeng, X. Mu, Y. Wang, and Y. Yu, “Multi-Designated Detector Watermarking for Language Models,” IACR ePrint 2024/1492, 2024. <https://eprint.iacr.org/2024/1492>

- [15] Y. Liu, X. Zhao, C. Kruegel, D. Song, and Y. Bu, “In-Context Watermarks for Large Language Models,” ICLR, 2026. <https://arxiv.org/abs/2505.16934>
- [16] J. Zhang, Y. Zeng, Y. Huang, Z. Ouyang, H. Chen, and X. Hu, “SemTrace: Source-Grounded Semantic Signatures for Tracing LLM Exposure to Protected Documents,” arXiv:2608.29575, 2026. <https://arxiv.org/abs/2608.29575>
- [17] A. Adelsbach, S. Katzenbeisser, and A.-R. Sadeghi, “Zero-Knowledge Watermark Detection and Proof of Ownership,” in Information Hiding, pp. 273-288, 2001. https://doi.org/10.1007/3-540-45496-9_20
- [18] Coalition for Content Provenance and Authenticity, “C2PA Technical Specification,” version 2.4, 2026. https://spec.c2pa.org/specifications/specifications/2.4/specs/C2PA_Specification.html
- [19] K. Yoo, W. Ahn, J. Jang, and N. Kwak, “Robust Multi-bit Natural Language Watermarking through Invariant Features,” ACL, 2023. <https://aclanthology.org/2023.acl-long.117/>
- [20] A. Hou et al., “SemStamp: A Semantic Watermark with Paraphrastic Robustness for Text Generation,” NAACL, 2024. <https://aclanthology.org/2024.naacl-long.226/>
- [21] J. Ren et al., “A Robust Semantics-Based Watermark for Large Language Models Against Paraphrasing,” Findings of NAACL, 2024. <https://aclanthology.org/2024.findings-naacl.40/>
- [22] H. Zhang, B. Edelman, D. Francati, D. Venturi, G. Ateniese, and B. Barak, “Watermarks in the Sand: Impossibility of Strong Watermarking for Language Models,” ICML, 2024. <https://proceedings.mlr.press/v235/zhang24o.html>
- [23] X. Chen, X. Li, Y. Xie, and Q. Long, “Selective Disclosure Watermarking for Large Language Models,” arXiv:2607.05353, 2026. <https://arxiv.org/abs/2607.05353>
- [24] G. K. R. Lau, X. Niu, H. Dao, J. Chen, C.-S. Foo, and B. K. H. Low, “Waterfall: Scalable Framework for Robust Text Watermarking and Provenance for LLMs,” EMNLP, 2024. <https://aclanthology.org/2024.emnlp-main.1138/>
- [25] D. Chaum and H. van Antwerpen, “Undeniable Signatures,” CRYPTO ’89, LNCS 435, pp. 212-216, 1990. <https://chaum.com/wp-content/uploads/2022/01/Chaum-Antwerpen-1990-Chapter-Undeniable-Signatures.pdf>
- [26] M. Jakobsson, K. Sako, and R. Impagliazzo, “Designated Verifier Proofs and Their Applications,” EURO-CRYPT, 1996. <https://markus-jakobsson.com/papers/jakobsson-eurocrypt96.pdf>
- [27] S. Saeednia, S. Kremer, and O. Markowitch, “An Efficient Strong Designated Verifier Signature Scheme,” ICISC, 2003. https://doi.org/10.1007/978-3-540-24691-6_4
- [28] L. El Aimani, “On Generic Constructions of Designated Confirmer Signatures,” IACR ePrint 2009/403, 2009. <https://eprint.iacr.org/2009/403>
- [29] H. Krawczyk, M. Bellare, and R. Canetti, “HMAC: Keyed-Hashing for Message Authentication,” RFC 2104, 1997. <https://www.rfc-editor.org/rfc/rfc2104.html>
- [30] T. P. Pedersen, “Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing,” CRYPTO ’91, LNCS 576, pp. 129-140, 1991. https://doi.org/10.1007/3-540-46766-1_9
- [31] S. Haber and W. S. Stornetta, “How to Time-Stamp a Digital Document,” Journal of Cryptology, vol. 3, pp. 99-111, 1991. <https://doi.org/10.1007/BF00196791>
- [32] C. Adams et al., “Internet X.509 Public Key Infrastructure Time-Stamp Protocol,” RFC 3161, 2001. <https://www.rfc-editor.org/rfc/rfc3161.html>
- [33] B. Laurie, E. Messeri, and R. Stradling, “Certificate Transparency Version 2.0,” RFC 9162, 2021. <https://www.rfc-editor.org/rfc/rfc9162.html>
- [34] H. Birkholz et al., “An Architecture for Trustworthy and Transparent Digital Supply Chains,” IETF SCITT Internet-Draft, work in progress, 2025. <https://datatracker.ietf.org/doc/draft-ietf-scitt-architecture/>
- [35] Coalition for Content Provenance and Authenticity, “C2PA Soft Binding API,” version 2.4, 2026. <https://spec.c2pa.org/specifications/specifications/2.4/softbinding/Decoupled.html>
- [36] L. Moreau and P. Missier, eds., “PROV-DM: The PROV Data Model,” W3C Recommendation, 2013. <https://www.w3.org/TR/prov-dm/>

- [37] R. Sanderson, P. Ciccarese, and B. Young, eds., “Web Annotation Data Model,” W3C Recommendation, 2017. <https://www.w3.org/TR/annotation-model/>
- [38] Crossref, “References and Relationships,” metadata schema documentation. <https://www.crossref.org/documentation/schema-library/markup-guide-metadata-segments/relationships/>
- [39] DataCite Metadata Working Group, “DataCite Metadata Schema,” version 4.7, 2026. <https://schema.datacite.org/meta/kernel-4.7/>
- [40] D. Shotton, “CiTO, the Citation Typing Ontology,” Journal of Biomedical Semantics, vol. 1, suppl. 1, 2010. <https://doi.org/10.1186/2041-1480-1-S1-S6>
- [41] M. Daquino et al., “The OpenCitations Data Model,” ISWC, 2020. <https://arxiv.org/abs/2005.11981>
- [42] Schema.org, “citation,” property definition. <https://schema.org/citation>
- [43] M. Nottingham, “Web Linking,” RFC 8288, 2017. <https://www.rfc-editor.org/rfc/rfc8288.html>
- [44] M. Nottingham, “Well-Known Uniform Resource Identifiers,” RFC 8615, 2019. <https://www.rfc-editor.org/rfc/rfc8615.html>
- [45] M. Koster et al., “Robots Exclusion Protocol,” RFC 9309, 2022. <https://www.rfc-editor.org/rfc/rfc9309.html>
- [46] J. Howard, “The /llms.txt File, v2,” community proposal, 2026. <https://llmstxt.org/>
- [47] A. Rundgren, B. Jordan, and S. Erdtman, “JSON Canonicalization Scheme,” RFC 8785, 2020. <https://www.rfc-editor.org/rfc/rfc8785.html>
- [48] M. Jones, J. Bradley, and N. Sakimura, “JSON Web Signature,” RFC 7515, 2015. <https://www.rfc-editor.org/rfc/rfc7515.html>
- [49] A. Backman, J. Richer, and M. Sporny, “HTTP Message Signatures,” RFC 9421, 2024. <https://www.rfc-editor.org/rfc/rfc9421.html>
- [50] R. Polli and L. Pardue, “Digest Fields,” RFC 9530, 2024. <https://www.rfc-editor.org/rfc/rfc9530.html>
- [51] Unicode Consortium, “Unicode Standard Annex #29: Unicode Text Segmentation.” <https://www.unicode.org/reports/tr29/>
- [52] X. Feng, Y. Zhang, H. Zhang, L. Y. Zhang, and S. Pan, “Tracing Provenance and Detecting Tampering with Complementary LLM Watermarks,” arXiv:2608.12713, 2026. <https://arxiv.org/abs/2608.12713>