Authenticity and Revocation of Web Content using Signed Microformats and PKI

Henrich C. Pöhls
University of Hamburg, Dept. of Informatics
Research Group Security in Distributed Systems (SVS)
Vogt-Kölln-Str. 30, D-22527 Hamburg, Germany
poehls@informatik.uni-hamburg.de

February 5, 2007

Abstract:
Semantically annotating web content will ease its extraction and processing by third parties. But with this processing destroys the context of the original publication. We show that because of the loss of this context information the quality of the web content is diminished. In this work we propose a Microformat to store digitally signatures. Signed micro content preserves the content's context and allows viewers to verify the origin and integrity of the content even after processing by third parties.

We further use existing methods from Public Key Infrastructures (PKI) to allow authors to revoke their consent to the publication of content. Using signed micro content this content revocation is detectable by viewers also after processing by third parties. While offering new control capabilities for authors, we still allow content to be free, unlike in some Digital Rights Management (DRM) approaches. We also shortly explain why our approach is beneficial for all the involved parties.

Zusammenfassung:
Semantisch annotierte Inhalte im Web erlauben eine einfachere, automatische Extraktion und Weiterverabeitung der Inhalte durch Dritte. Aber diese Extraktion von sog. Micro Content zerstört den Zusammenhang (Kontext), in dem die Inhalte ursprünglich standen. Wir zeigen, welch wichtige Informationen hierdurch verloren gehen und wie sich dadurch die Qualität des extrahierten Inhaltes verringert. Zur Lösung schlagen wir daher sog. Signed Micro Content vor und beschreiben ein sog. Microformat, welches Digitale Signaturen speichert und semantisch auszeichnet. So signierter semantischer Web-Inhalt erhält die Kontext Information und erlaubt es den Betrachtern, Ursprung und Integrität der Inhalte, auch nach der Verarbeitung durch Dritte, zu überprfen.

Ausserdem zeigen wir, wie existierende Methoden aus dem Umfeld von Public-Key-Infrastrukturen (PKI) dazu genutzt werden können, um sich als Author von Inhalten zu distanzieren. Diese Inhalte Revokation kann von anderen Nutzern durch die Nutzung von Signed Micro Content auch bei bereits weiterverarbeiteten Inhalten nachvollzogen werden. Wir geben hiermit Autoren neue Werkzeuge an die Hand, um Inhalte besser zu verwalten. Wir schränken aber gleichzeitig nicht den Informationsfluss und den Zugang zu den Inhalten ein, wie es bei den meisten Vorschlägen zum Digital Rights Management (DRM) der Fall ist. Inhalte bleiben frei zugänglich, dennoch zeigen wir warum es sich für alle Beteiligten lohnt, Signed Micro Content zu verwenden.

1 Introduction

Web content is getting easier to understand by machines by means of semantic annotations [1] [2]. More and more services harvest, process and re-use web content to offer new services to the users. Processing content from different data sources is one functionality of the "phenomenon" called Web 2.0 [3]. Two very simple uses of published content by third parties are indexing and caching as done by web search engines; and the aggregation of content from different sources by so called aggregators.

Content is extracted from the authors web page and, after processing, re-displayed. After this all the information that was originally deductible from browsing the author's web page is lost. We will call this information *context*. Additionally the original author is often not immediately verifiable, and viewers do not know, if the author still consents to the content (i.e. is it still as such on the author's web page).

This work will examine the consequences suffered from extracting and processing semantically structured content. We will then present means to preserve the context. Our approach offers new capabilities to authors and viewers (i.e. the human end-user's of content) of web content. Among these are integrity protection, origin verification, author consented processing and perhaps most notably revocation of authentic content. Finally we will introduce an addition to existing semantic annotation done with Microformats [2] by defining a Microformat that carries all the signature related information.

Compared with existing solutions like Digital Rights Management (DRM), that offer strong content protection within closed systems, our solution still allows free access to information in an open environment as our approach is not trying to protect the confidentiality. As such our approach does not limit access to information and integrates well with existing applications.

1.1 Paper outline

Our paper is organized as follows: In section 2 we will start defining some terms to have a common understanding. We then use a scenario in section 3 to outline the problems authors, and viewers alike, face when publishing or viewing micro content. In section 4 we present our solution and show how authors can add cryptographically protected semantic information allowing viewers to validate their authorship, the processor's compliance to the processing rules, and the author's consent to publication. We also show in more detail how the verification procedures work and how consent to content publication can be revoked. After a brief discussion of our approach, also showing the benefits for all participants, we turn to related work in the field in section 5. And finally in section 6 we conclude.

2 Definitions

In this section we will define certain terms used throughout the paper. Often these terms are not new but we would like to clarify from the beginning what we mean by those terms.

2.1 Definition: Micro content

We will use the term *micro content* for content that contains semantically and logically structured content. It is semantically annotated in such a way that it is not only human-readable, but machine-readable as well. One micro content covers one semantical concept. As such micro content can be nested in, included, linked to or otherwise related to other micro contents. Micro content in itself, is structured. We will call the sub-structures of micro content *sub-properites*. Each sub-property is addressable and identifiable.

Semantic annotations can be done using Microformats [2] for HTML based web content or using RDFa [1] in the more general "Semantic Web" [4]. Our general concept is applicable universally to all forms of content, but we will use web content (in HTML or XHTML) as our main example for simplicity. Further we will base our examples on Microformats annotation allowing easy applicability. Microformats use the HTML `class` attribute to store annotations. See [2] for more details.

To make it more comprehensible figure 2.1 shows an excerpt of a semantically annotated web page. With Microformat annotation the telephone number (`class="tel"`), the email address (`class="email"`) and the full name (`class="fn"`) form our office's machine-readable contact information (`class="vcard"`).

```
1  <div class="vcard">
2   <h1 class="fn">SVS - Office</h1>
3    <a href="mailto:svs-office@informatik.uni-hamburg.de" class="email">Email us.</a><br/>
4          Tel.: <span class="tel">+49 40 42883 - 2510</span><br/>
5       Fax: +49 40 42883 - 2086<br/>
6       Room: F-631<br>
7  </div>
```

Figure 2.1: Semantically annotation using the Microformat *hcard*

From figure 2.1 it is quite obvious that a single web page can contain several semantically annotated pieces of micro content. For them to be separately addressable we assume they have a different Uniform Resource Identifier (URI). For example `http://www.domain.org/page.html#1st` and `http://www.domain.org/page.html#2nd`.

2.2 Definition: Involved parties

In general we see three distinguishable parties: First there are the *content authors*, who are the original creators of content, secondly *content processors* which process original content and finally we will call the consumers of both original and processed content *viewers*.

2.3 Definition: Micro content processing

We will use the term *micro content processing* or *content processing* to describe the manipulation done on micro content by third parties. This includes, but is not limited to, re-publishing the content (eg. storing, mirroring and caching). Additionally parts of the content can be omitted (eg. excerpting, citing or aggregating). Content might also be re-ordered during the processing. We assume that due to the open nature and versatility of the World Wide Web (WWW) the author will not be able to foresee all future processing his content might be subjected to after initial publication.

2.4 Definition: Context

By *context* we mean all the surrounding elements that were present when the original content, as authored by the author, was initially published on the web, and could be directly consumed by viewers. One very simple example of context would be the server the content was originally published on. For simplicity we further reduce this to the full Universal Resource Locator (URL). URLs are used by humans to identify the content's source. The URL of a web page is also used for identifying the author. Phishing attacks [5] build upon the use of URLs for source identification by using look-alike URLs. Additional means of source identification, like an originally SSL secured connection with the server, are also part of the context.

We would like to state one problem we see in micro content processing upfront: Re-displaying processed content destroys the content's original context, more in section 3.

2.5 Definition: Consent

With *consent* we mean the author's agreement to the processing of her content and also her agreement to re-publication. In the case of storing/caching this also includes a notion of "freshness", allowing authors to disagree with an earlier statement, but on the other hand allowing viewers to retain the proof that at a given time in the past the content was indeed representing the author's view (and was published with her consent).

3 Problems of micro content processing

In this chapter we will identify the security problems related to processing of micro content from the author's and viewer's point of view. We will start with a small scenario:

Assume we take a single address information as our micro content (like the example in figure 4.3). A good example for semantically annotated addresses would be a yellow page listing [6] or the "About me" sections of Blogs. Annotated content now enables machines to understand the content as addresses and therefor third party services exists that can harvest and further process it. In our scenario we have an aggregating service that processes address data and composes a searchable telephone directory. First demos [7] of such services already exist.
This concludes our scenario, we will now look at the most interesting open questions in more detail.

3.1 Origin authentication

As already stated earlier, one of the major drawbacks of processing micro content is the loss of context. Without the initial context the only way to verify the origin is to follow a link back to the original source of publication (also called perma-link or back-link and observable in many Blogs). Of course, for this to work such links need to be kept and the original content needs to be still available. And in order to verify this the user has to follow that link and check and find the micro content. This is no automated process, takes additional time, and is prone to human errors.

In our scenario:
The user named Alice issues a search request for the phone number of her bank's call-center. Assume the search returns three hits, which telephone number would Alice call to be assured to talk to an clerk from her bank?

Manually verifying which information is on the author's website not only takes considerably more time and is prone to human errors, but it also diminishes the usefulness of the micro content processing services used to find the information in the first place.

3.2 Consent to content processing

The context of a web page is also used to judge the author's consent with the content published.

An example: Content that is displayed on a page with the following URLs[1]:

- *news.bbc.co.uk/2/hi/technology/5384170.stm*
- *www.microsoft.com/technet/security/advisory/925568.mspx*
- *msinfluentials.com/blogs/jesper/archive/2006/09/19/Block-VML-Zero_2D00_Day-Vuln-on-a-domain.aspx*

might be judged differently according to the viewer's assessment of the source's credibility and previous experience. With only the URLs as context viewers would implicitly take information published on www.microsoft.com as a statement Microsoft agrees to, whilst reports by third parties, like the BBC, about Microsoft might not be agreed to by Microsoft.

Even if the author is verifiable, the processing (nearly always) happens without the original author's knowledge and control. Thus their content is processed without their explicit consent. Todays author's control over the re-use of their web content is limited to either restricting the very first access, by means of access control, or having no control once the content is published.

Another closely related problem is the content's integrity. Here integrity is used in a more semantical sense. During content processing not only the original context is lost but also part of the original content might get changed. For example citing or aggregation content removes part of the original content. This is often useful and desired, but as authors might want to retain control over what processed content is still associated with them as authors and what shall no longer be associated with them.

Shortly exemplified:
Content might be cited in such a way that the newly "generated" statement is misleading, it can be misquoted. For some content this could be greatly reduced if only full sentences would be cited. In this case we would like to hand authors the tools to state that they only want to be named the "author" if the citation obeys their rule.

As such, origin authentication shall only be valid if the micro content is processed in accordance with author defined rules. We would like to stress the point that this is different from the protection classical DRM solutions are desired to offer. In our approach full access to the content by third-parties is retained, only falsely claiming that the author consented to this is restricted. To make it even clearer: Copy and paste from the author's content is still possible for any third-party.

[1] These URLs are just randomly chosen URLs that all cover a recent security-related news story. The author has no affiliation with the page authors, nor assessed their suitability or credibility in this matter or another.

3.3 Revoking consent to content publication

The re-publication of processed content will spread (at least in parts) the author's original content and thus make it visible to more viewers. On the other hand, as this happens without the knowledge of the author, viewers might view processed content that is processed without the author's consent, thus, might be less reliable or outdated. For viewers of processed content there is no standardized way to check if it is re-published with the author's consent and if the content was processed in accordance to the author. For authors it is hard to find content that got processed without their consent. If authors find their content and do not agree either with the way the content was processed or no longer agree with their content itself, they have no standard process to express this to the third-party. There is no standard for content revocation on the web.

4 Solution: Cryptographically signed micro content

Our approach is to cryptographically sign the content itself using a digital signature. Thus the origin can by authenticated via authenticating the public-key used for signing. Our signature is generated over a hash tree over the micro content's sub-properties. This hash tree is comparable to a Merkle hash tree [8]. Now processors can omit certain content, whilst retaining a valid author generated signature. This allows authors to express consent to only certain content processing without a-priori knowledge of the processing and without further actions on the author's side. Additionally our approach reuses existing methods for certificate revocation to revoke the consent to publication even after the content is processed.

Instead of restricting content processing in general by introducing another form of DRM, we see an increased need to provide a technical solution to keep the contents' context intact and verifiable for viewers of processed content.

4.1 Microformat for digital signatures

Using the already existing approach to semantically markup content using Microformats, we propose a new Microformat to specify the information related to digital signatures. We implemented a prototype using Microformats as micro content format. We have chosen Microformats, because it allows easy addition of semantic information to existing HTML-based structured content. Several content creation tools already allow to generate Microformatted content. Thus, we believe the signature generation and verification can also be build into the same tools, allowing to add additional value by signing the content.

The structure and names are chosen in the style of XML Signature [9], the existing XML structure for digital signatures. Figure 4.1 shows the proposed structure. To better understand Microformats in general: Any HTML structure allows the `class` attribute to have one or more values. As such `class` can be used to store also annotations, so below is just one way of representing the values needed for the cryptographic information by using the HTML elements `div`, `a`, `span` and `abbr`. The latter is parsed according to Microformats in such a way that the value is taken from the `abbr`'s `title`-value.

There are several differences compared to the original XML Signature structure. Most notably the proposed Microformat takes a less nested approach. For example instead of putting `digestmethod` and `digestvalue` into a sub-container called `reference` we flattened the structure. Also we added fields that would allow more room for capturing different signature formats. For example additionally to flattening `keyinfo` we added `keyvalue`, which allows to store information about the key.

```
<div class="hsig">
  <a class="canonicalizationmethod url" href="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"></a>
  Signed using <span class="signaturemethod">RSA</span>.
  Hashed using <span class="digestmethod">SHA1</span>.
  <abbr class="manifest" title="vcard:fn,vcard:email">Name and eMail signed.</abbr>
  <abbr class="digestvalue" title="57c8105e6d944[...]a14c4cea7f53">The Hash.</abbr>
  <abbr class="signaturevalue" title="7m6NS6ANCa[...]Kl42Rr+Pfw==">The signature.</abbr>
  <abbr class="keyinfo" title="X509">I have an X509 certified key.</abbr>
  <abbr class="keyvalue" title="-----BEGIN CERTIFICATE----- E693c4[...]
           [...]MIICIzCCAc2gAgANB-----END CERTIFICATE-----" >
       My X509 certificate containing my public-key.
  </abbr>.
</div>
```

Figure 4.1: hsig: Microformat for Digital Signatures with example values

This Microformat could be used in conjunction with other already defined Microformats like hcard, hcalendar or hreview. As our proposed Microformat for digital signatures is content agnostic it can be used to build compound signed Microformats from all existing and future Microformats. To bind the signature to an existing Microformatted micro content we foresee three possibilities, that are again conforming to the use of XML Signatures: *Enveloping*, *enveloped* and *detached* signatures. We will shortly describe the three cases.

4.1.1 Enveloping signature

For an enveloping signature the `hsig` container contains the Microformatted micro content that is signed. The micro content's sub-properties are referenced in the manifest using the Microformat's name followed by the sub-property, separated by a colon (e.g. `vcard:fn`).

4.1.2 Enveloped signature

In the case of an enveloped signature the `hsig` container is contained within the Microformatted micro content that is signed. Again the micro content's sub-properties are referenced in the manifest using the Microformat's name followed by the sub-property, separated by a colon (e.g. `vcard:fn`). Figure 4.3 shows this case.

4.1.3 Detached signature

This case is needed when a signature shall cover more than one micro content from that page. The micro contents that are part of the signature are then referenced using an `<object>` or `<a>` HTML element inside the `hsig`-section. In this case the micro content needs an `id`, which is then used for reference instead of the Microformat's name. This case is the most complex one and is not further elaborated for brevity.

4.1.4 Manifest details

In our prototype the manifest is stored under the class name "manifest" (see fig. 4.3). To express the author defined rules for content processors we use an expression based on the extended Backus-Naur-Form (EBNF). The manifest first identifies for which micro content structure it will define processing restrictions. As the manifest is always embedded into the signed micro content itself this aids parsing, especially if a micro format contains several sub micro formats (see fig. 4.2). After the micro format identifier the constraints over sub-properties are expressed using the following EBNF symbols and conventions. We use "," as concatenate-symbol, "|" as separator-symbol, "(" and ")" for grouping symbols, and "[" and "]" for optional symbols. The sub-property identifiers are the terminal symbols.

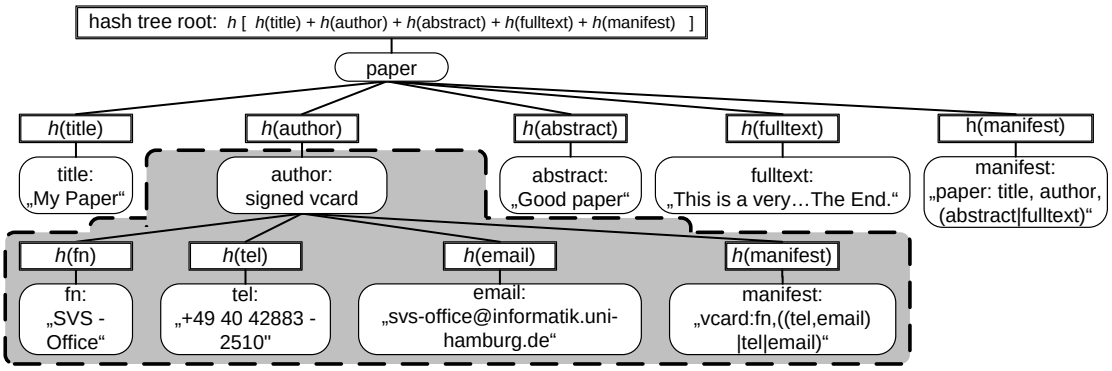

Figure 4.2: Hash and data tree of a paper, containing itself a signed address micro content

In general the manifest shall non-ambiguously articulate the tree's structure required to build the hash tree. The manifest `paper: title, author, (abstract|fulltext)` restricts valid processing to omissions of only the abstract or the full text. If content processors omit an original value, such as the full text, the original value must be replaced by its substitution hash. We use an additional sub-property named `sig-substitute`, that lists all the left-out sub-properties' identifier and their hash values. But other solutions are possible and will be explored in future versions of the prototype.

During signature verification viewers check the integrity and compliance to the processing rules set forth by the author as described above. Only the author can set the rules as the manifest is protected against unauthorized changes as a part of the digitally signed micro content. Additionally our proposed signature generation and verification algorithms do not allow to substitute the manifest with a hash.

4.2 Signature generation by the author

In our approach the author generates the signature over micro content along the following steps:

1. Choose sub-properties to be omitted, and under which circumstances.

2. According to this decision, write the *manifest*. The manifest defines the valid processing using a form of EBNF. Valid processing shall not invalidate the author's signature, thus a valid signature expresses the author's consent to the occurred processing. Section 4.1.4 gives more details.

3. Add the manifest itself as the first item to a list. This list contains all the sub-property that are hashed.

4. All the signature related sub-properties that need to be protected by the signature are also added to the list. E.g. algorithm information or links to the `microcontent-revocation-list`(MRL). See section 4.3.3 for more details on micro content revocation.

5. Add all the sub-properties from the manifest to the list. The list is now complete and contains all items that can be protected by the signature.

6. Hash all list items individually using a cryptographically secure hash function. In our prototype we used SHA1.

7. Build a tree of the hashes according to the manifest. All hashed sub-properties listed in the manifest are added as leaves. If they are further structured the tree is further expanded, until all the leaves contain the hash of a sub-property. This hash tree is based on a Merkle hash tree [8]. Thus, the parent node's value is a hash of the hashes of its child nodes.

8. Compute the hash tree root and sign it, using a secure digital signature algorithm. In our prototype we used RSA for digital signatures.

9. Add the root's digest value and the signature as sub-properties to the micro content. Finally, the author can add additional signature related information. One important information that can be added is a public-key certificate containing the author's public-key corresponding to the secret-key used for signature generation.

With this information added the micro content is transformed into *signed micro content*. The content original structure is preserved and still freely accessible, but viewers and processors can now check the signature.

Figure 4.3 shows a possible implementation approach for embedding signed micro content in HTML using our proposed Microformat `hsig`. The micro content is based on the Microformat "hcard" already shown in 2.1. The class names carrying the sub-properties with cryptographic values are underlined.

4.3 Verification of signed micro content by viewers

After showing how the signature was generated we will look at the verification process. The processed content is viewed with a "content browser" capable of checking the validity of signed micro content. The viewer's browser is trusted to perform the necessary steps.

```
 1 <div class="vcard">
 2   <h1 class="fn">SVS - Office</h1>
 3   <a href="mailto:svs-office@informatik.uni-hamburg.de" class="email">Email us.</a><br/>
 4     Tel.: <span class="tel">+49 40 42883 - 2510</span><br/>
 5     Fax: +49 40 42883 - 2086 <br/>
 6     Room: F-631 <br>
 7   <div class="hsig">
 6     <abbr class="manifest" title="vcard:fn,((vcard:tel,vcard:email)|vcard:tel|vcard:email)">
         This
       </abbr>
 7     <abbr class="digestvalue" title="57c8105e6dd944[...]a14c4cea7f53">
         content
       </abbr>
 8     <abbr class="signaturevalue title="7m6NS6NCa[...]Kl42Rr+Pfw==">
         is signed.
       </abbr>
 9     <abbr class="digestmethod" title="SHA1">It was hashed using SHA1.</abbr>
10     <abbr class="signaturemethod" title="RSA">And signed using RSA.</abbr>
11     <abbr class="keyvalue" title="-----BEGIN CERTIFICATE-----
                    MIICIzCCAc2gAgANB[...]-----END CERTIFICATE-----" >
12               My <span class="keyinfo">X509</span> public-key certificate.
       </abbr>.
13   </div>
14 </div>
```

Figure 4.3: Signed micro content using an enveloped *hsig* Microformat

This browser does not need to be a web browser. The verification process is threefold allowing for origin authentication, consent to processing validation and last but not least content revocation.

4.3.1 Origin authentication

Viewers can verify the origin by verifying the signature and the authenticity of the public-key corresponding to the private-key used for generating the signature. They can do this by acquiring the public-key out of band or if a Public Key Infrastructure (PKI) is available verify a public-key certificate that was provided. Thus, they are able to verify the author (via certificate verification). In our prototype we used X509 [10] certificates, but our proposed solution is not restricted to this. For example, the pretty good privacy (PGP) approach with a web of trust can also be used to verify the author.

4.3.2 Validation of consent to processing

Verification of the signature implicitly yields that the integrity was not violated and that the rules set forward in the manifest still apply to the content remaining after processing. By this the viewer is able to verify the author's consent to the processing that the content might have undergone. Viewers can further detect which sub-properties of the content have been omitted and try to find missing values semi-automatically.

4.3.3 Revocation of consent to processing

To reduce confusion beforehand: We will use certificates different from how they are used in the PKI world. Especially we will add a different semantic meaning to certificate revocation.

Micro content revocation verification works as follows: The verifier first checks the validity of the authors public key certificate. If the certificate is invalid or revoked this clearly indicates a revocation due to problems relating to the author's private-key. There are several reasons for that key compromise, lost key, no more funding to pay a CA for a renewal of the public-key certificate, etc. It is important to notice that this relates to the public-key certificate. We propose another type of certificate called *micro content certificate* (or shorter: content certificate). This certificate is issued by the author and vouches for the author's consent to the micro content. It resembles in many facets the already existing public-key certificates used in PKI. In fact we propose using the same technical mechanisms and will continue to work on prototypes implementations showing the possibility to reuse existing PKI components. Instead of carrying a public-key it will carry information identifying the content, such as the hash tree's root value.

With micro contents certificates authors can assign lifetimes (not-valid-before and not-valid-after) for their content such as with public-key certificates. If the actual date is not within the lifetime restrictions set forward by the author the consent to processing is revoked and verification will fail. Authors could also manage a signed list of revoked micro content, a *micro content revocation list (MRL)*. Where such a list can be obtained is stated, as with usual certificates, inside the certificate. If a MRL can be obtained the verifier will check the list's validity by signature verification. The MRL is signed using the author's key. If the inspected signed micro content is listed the micro content certificate verification will fail. A future option is to re-use the functionality offered by the Online Certificate Status Protocol (OCSP) [11].

A failure of these content certificate checks indicates that the author no longer "supports" the micro content and thus has revoked the consent to processing. In this case the viewers' browsers might adhere to user-defined policies. This could result, for example, either in not displaying this micro content at all, or marking it as "revoked".

We used X.509 certificates as content certificates in our prototype. Mechanisms for X.509 certificate revocation are standardized. Checks for revoked certificates are implemented into browsers already. Our approach can be build upon those mechanisms allowing easy adoption.

Our approach allows authors to change their expression of consent over time, even after content is processed. A priori, authors control the life time of their consent either by the validity period of a public-key certificate or the validity period stated in the micro content. For example stating that an offer is only valid for the next two weeks, but afterward could be kept on record for a pricing history. A posteriori control is managed through revocation lists. Authors can also enhance the life time by issuing new certificates with longer validity periods.

4.3.4 Verification steps

The signature verification needs to re-compute the hash tree root according to the tree's structure and processing rules defined in the manifest. Broken down into steps, the signature verification works as follows:

1. Verify the validity of the public-key. For example using out-of-band verification mechanism or a public-key certificate.
2. Construct the hash tree from the manifest.
3. Re-compute the hash tree root:
   a) Try re-computing hashes of each sub-property listed in the manifest.
   b) If there is no sub-property value, check if the manifest allows omission.
   c) If omission is allowed, take substitution hash, if not, abort with failure.
   d) If no sub-property and no substitution hash is found, abort with failure.
4. Check the signature: If the re-computed hash tree root matches the signed root the signature verification is successful, thus the origin can be authenticated and the author's content-processing rules where respected. Otherwise both, origin authentication and consent-to-processing-validation, indistinguishable from each other fail.
5. Check the micro content certificate:
   a) Verify the validity of the certificate by verifying the certificates signature. For example using the out-of-band gained author's public-key.
   b) Verify the content certificate's lifetime.
   c) Check if the content certificate is revoked by comparing it to the latest MRL or by the use of other services.
6. If the content certificate is valid the author's consent to the processed content is still valid. Otherwise the consent has been revoked, and the content certificate verification fail.

Table 4.1 shows the four possible interpretations according to the results (pass or fail) of two of the three checks. The "standard" verification of the author's public key is omitted for brevity. In all cases the viewer can still view and extract the content, this is different in most DRM systems. In all cases except the double failure (case 3) signed Microcontent provides added value to the viewers. Case 0 obviously indicates the positive result for the viewer. The other two cases provide more or less useful information about the content and its processing. It can be used for historical purposes (case 2) or to alert the author or the community of unwanted processing (case 1). If the processing service used adheres to a policy forbidding re-publishing such content, than case 2 shall alert the author or the community of a misbehaving service.

	Signature Verification obeying manifest passed	**Signature Verification obeying manifest failed**
Content Certificate verification passed	Case 0: consent expressed and processed in consent	Case 1: consent expressed, but unwanted processing
Content Certificate verification failed	Case 2: consent revoked, but processed in consent	Case 3: consent revoked and unwanted processing

Table 4.1: Interpretation after the two verification checks

4.4 Processing signed micro content

Now that we have stated how micro content is signed and verified we will turn to the third party, the content processors. Authors defined the processing they consent to by means of grouping the sub-properties in the manifest. As this is also part of the hash tree and as such protected by the author's signature. But as we are using a hash tree based signature, a sub-property value can be substituted with the hash of that value. We will refer to this as a `substitution-hash`. Thus it is possible to omit content without invalidating the signature. If the content processor re-publishes the substitution hashes together with the not omitted sub-properties, the viewer can re-compute the hash tree's root and validate the signature.

So valid-signature-retaining processing can be described by the following steps:

1. First, check the validity of the signature. If invalid there is something wrong with the content. In this case the processor might adhere to a policy that would stop him from processing and re-publishing this micro content after this first step.

2. If the certificate is invalid or revoked, the author might no longer actively "support" the content. In this case the processor might adhere to a policy that would stop him from processing and re-publishing this micro content after this second step.

3. Decide which micro content sub-properties the processor wants to omit.

4. Check the author's manifest if the desired content processing is within the guidelines. If it is within the guidelines valid-signature-retaining processing can continue.

5. For each sub-property that is omitted generate a hash over that sub-property (the substitution hash). If the processor wants to omit the content of multiple sub-properties, the processor can check whether or not this multiple sub-properties are a complete part of a hash tree. If the latter is true, it can subsidize the multiple hashes with just one hash.

6. Add the needed substitution hashes into the processed micro content.

7. Finally, publish the processed micro content preserving a valid signature.

In our simple example (see fig. 4.3) the list of sub-properties is quite small. Consequently, we rather have a list of hashes as the height of the tree is just one. Never the

less, a content processor can remove the telephone number, to keep the signature valid he would insert for example the following as sub-property of the signature (between line 7 and 12): `<abbr class="substitutehash" title="tel:a68fd14[...]c161f8fe">some parts removed</abbr>` If micro contents become more complex and nested (see fig. 4.2) the benefits of the hash tree based approach become more visible.

4.5 Enforcement of author defined content processing restrictions

Signed micro content, as we propose it, imposes no viewing or re-publication restrictions known from digital rights management (DRM) systems often mentioned in the context of content publication. Thus, our scheme does not guarantee the enforcement at the processors nor at the viewers. Rather we envision that users will appreciate the added value they gain with signed micro content and thus, demand it. As with the semantic web, it is the small steps like Microformats that allow early adoption. Nevertheless, once viewers become aware of the benefits and authors get some control over content processing, demand could rise and create a positive feedback loop.

At first sight this benefits viewers and authors a lot more than the processors. On second thought, it is at the viewers discretion to decide which processing service to use. DRM like methods are not yet (or never will be) suitable in an Internet scenario. We also believe that users want to get their data not only "quickly and easily" but that they want quality and authentic data.

A simple scenario:
We have two content processors A and B. Both offer a searchable address database built from harvested micro content. Processor A adheres to a self-given policy that it will only process micro content according to the rules and thus, the author's signature is still valid. Every night they re-check all their stored micro content, new and old. They will remove any content that fails signature or content certificate verification. Content processor B, on the other hand, eagerly harvests all the content he gets and processes it without any respect to the manifest. Processing service B would perhaps find multiple hits, including old and no longer maintained ones. Being able to verify micro content allows viewers to make more informed decisions.

On the Internet users need to verify the authenticity of viewed content. Signed micro content allows this and adds value to the micro content. Viewers will, more often, use content processors that process signed micro content according to the author defined rules. The example given shows two extremes only, but highlights that processors willing to adhere to a policy, and enforce author defined rules, will get more attention and therefore benefit from a positive feedback loop. Thus we believe in the "power of crowds".

Additionally policy-compliance can be checked by all viewers regularly using the service, thus processors are constantly monitored for their policy-compliance. This already leads us to the discussion of our proposed solution.

4.6 Discussion

Our approach could provide a new use case for digital signatures and introduces personal PKIs. It is far fetched to name the semantic web the killer application for digital signatures, but the existing concepts of PKI can be "abused" to enhance micro content and offer functionalities of content management such as unwanted-processing detection (see section 4.3.2) and content revocation (see section 4.3.3). Most obviously adding a signature adds to the overall size. Just a reminder, in this case "micro" has nothing to do with the size of the content. It rather states that it contains a single, minimal semantic concept. The data itself can be of arbitrary size. The actual increase depends on cryptographic parameters like hash size or an embedded certificate. We suppose the increase in size is marginally compared to today's bandwidth and communication costs. In total viewers might even save bandwidth, as they do not need to trace back to the content's origin. The advantages of the hash tree based signature become more visible if the micro content increases. In the small example from figure 4.3 the telephone number's hash is actually longer than the original value. But in the example depicted in figure 4.2 substituting the full text with the substitution hash value clearly shortens the total length of the processed micro content.

The cryptographic operations introduce another, more important, overhead in the processing. In general signature generation is more labor intensive than verification, it only has to be done once by content authors. Also the generation of new content-keys is costly. The costs of generating a new key for the author only needs to be done once. Then for each content we need two signatures one for the content's signature and one on the content's content certificate. But authors also gain the benefit of content processing control. Content processors, as the second player, will process a lot of micro content. Their computing costs are limited to the verification of signatures and the generation of hash values. For valid processing they only have to spent the small costs for hash computations. On the other hand processors can now offer services of increased value. Finally the viewers, they again only have to validate signatures, two for each content.

An example:
Assume one has written a paper and publishes it with semantic annotation. This annotation could contain the following sub-properties: Paper's title, information about the author, a short abstract, and a full text version. The author information could be a micro content like the address in our example code, and thus now adds to the tree's depth. This is shown in figure 4.2. In our example the author's details shall be re-published together with either the full text or the abstract. Imagine the above paper is co-authored by several. Each co-author signs her address, as micro content. This allows new applications by verifying the individual signatures and certificates. We encourage to nest signed micro content. Each micro content processor can also be author. For example, my paper (signed micro content) can be referenced to in another author's signed micro content.

5 Related work

The concept of hash trees is already used in content distribution. For example P2P systems use it to verify that downloaded parts are from the content that was requested. But in P2P systems the hash tree's root is not cryptographically protected.

Work done by Devanbu et al. [12] allows to verify the answer to a selection query over a XML document. They also use a Merkle hash tree to allow omission of parts of the XML document not asked-for in the query. Clients obtain the hash tree root from the authors in a secure way. They can then verify if the returned parts of the document are complete and correct. Thus, the author can be checked. But in their work authors have no control over the queries and thus, less control over the content returned. Our approach is different, as we give control to the authors what content can be omitted. Also our approach is more suitable in an web environment.

Bertino et al. [13] in 2004 and more recently in 2005 Carminati et al. [14] show how, again XML data, can be protected without trustworthy publishers. Their approach additionally employs confidentiality protection through encryption. Our approach focuses on Internet content that is created for publication, distributed and openly accessible, and as such confidentiality plays a minor role. But the main difference between their approaches and our proposal is our use of existing public key infrastructure mechanisms. We use them to revoke the author's consent to content publication done by third parties.

To the best of our knowledge no one has yet proposed the use of PKI mechanism to revoke signed web content after it has been processed and re-published by third-parties. Our approach is also different as it uses hash tree based signatures for semantically structured web content, allowing authors to control content omission, and giving web content processors the freedom to process while retaining a valid signature.

6 Conclusion and Future Work

To sum up, our approach hands the following features to content authors, processors, and viewers: First, viewers can validate the content's author and thus origin, after valid content processing. Second, the content's integrity, including sub-content dependencies, can be validated. Author compliant content processing is verifiable by viewers. Last but not least, and probably the most notably, the author's consent to the content can be revoked even after publication by third-parties. This revocation remains still visible for viewers after valid content processing.

We have given examples and argued that content processors are obliged to keep the signature related information with the content as it adds value for the viewers. Instead of stripping it off an intact signature enhances the quality of the content.

This functionality was not available on the web before. It enables new opportunities and new services. We achieve the above by adding a digital signature to each micro content and showed how existing PKI mechanisms like certificate revocation can be re-used. These technologies are already embedded in todays browsers and are already used on the Web. Thus we achieved our goals, without data lock-in, like in DRM, and without re-inventing the wheel.

Continuing the work on our prototype, we plan to make the functionality available to the community in the near future. This will allow us to integrate signed micro content even better into services for content creation and distribution, and allow us to explore new possible applications.

Bibliography

[1] W3C, "RDFa Primer," www.w3.org/TR/xhtml-rdfa-primer, May 2006.

[2] Microformats, "website," www.microformats.org, Aug. 2006.

[3] T. O. Reilly, "What is Web 2.0," www.oreillynet.com/lpt/a/6228, Sept. 2005.

[4] T. Berners-Lee, J. Handler, and O. Lassila, "The semantic web," *Scientific American*, May 2001.

[5] R. Dhamija, J. D. Tygar, and M. Hearst, "Why phishing works," in *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*. New York, NY, USA: ACM Press, 2006, pp. 581–590.

[6] M. of Posts & Telecom. Cambodia, "Cambodian yellow pages," www.yellowpages-cambodia.com, Aug. 2006.

[7] Technorati, "Microformats search," kitchen.technorati.com, Oct. 2006.

[8] R. Merkle, "A certified digital signature," in *Advances in Cryptology*, 1989.

[9] Eastlake, Reagle, and Solo, "Xml-signature syntax and processing. w3c recommendation," www.w3.org/TR/xmldsig-core/, Feb. 2002.

[10] R. Housley, W. Polk, W. Ford, and D. Solo, "Rfc 3280," Apr. 2002.

[11] R. Ankney, A. Malpani, S. Galperin, and C. Adams, "Rfc 2560 - x.509 internet public key infrastructure online certificate status protocol - ocsp," http://www.ietf.org/rfc/rfc2560.txt, June 1999.

[12] P. Devanbu, M. Gertz, A. Kwong, C. Martel, G. Nuckolls, and S. Stubblebine, "Flexible authentication of XML documents," in *8th ACM Conf. on Computer and Comm. Security*, 2001.

[13] E.Bertino, B.Carminati, E.Ferrari, B.Thuraisingham, and A.Gupta, "Selective and authentic third-party distribution of XML documents," *IEEE TKDE*, vol. 16, pp. 1263–1278, 2004.

[14] B. Carminati, E. Ferrari, and E. Bertino, "Securing XML data in third-party distribution systems," in *Proceedings of 14th ACM CIKM*, 2005, pp. 99–106.