The Eternal Resource Locator: An Alternative Means of Establishing Trust on the World Wide Web

Ross J. Anderson, Václav Matyáš Jr., Fabien A.P. Petitcolas
University of Cambridge Computer Laboratory
Pembroke Street, Cambridge CB2 3QG, UK
{rja14, vm206, fapp2}@cl.cam.ac.uk; http://www.cl.cam.ac.uk/~{rja14, vm206, fapp2}

Abstract

Much research on Internet security has concentrated on generic mechanisms such as firewalls, IP authentication and protocols for large scale key distribution. However, once we start to look at specific applications, some quite different requirements appear. We set out to build an infrastructure that would support the reliable electronic distribution of books on which doctors depend when making diagnostic and treatment decisions, such as care protocols, drug formularies and government notices. The integrity, authenticity and timeliness of this information is important for both safety and medico-legal purposes. We initially tried to implement a signature hierarchy based on X.509 but found that this had a number of shortcomings.

We therefore developed an alternative means of managing trust in electronic publishing. This has a number of advantages which may commend it in other applications. It does not use export-controlled cryptography; it uses much less computation than digital signature mechanisms; and it provides a number of features that may be useful in environments where we are worried about liability. We also present our intermediate solution – the first ever large scale deployment of one-time signature systems. The move to one-time signatures enabled considerable simplification, cost reduction and performance improvement. We believe that similar mechanisms may be appropriate for protecting other information that changes slowly and remains available over long time periods. Book and journal publishing in general appear to be strong candidates.

1 Introduction

Medicine is one field in which serious attempts are underway in a number of countries to build large-scale decentralised trusted systems over the Internet to support a number of aspects of patient care, administration and research. Medical informatics has made unique contributions to the general pool of security know-how, and as medical practice is highly decentralised, many of these lessons may be applicable to Internet applications in general. Examples include the following.

- Conventional security policy models like Bell-LaPadula and Clark-Wilson do not work well in medical telematics, as they assume that security administration is centralised. Medicine is a business in which the field operatives are the most highly trained, the most trusted by the clientèle, and in most countries are also legally burdened with the duty to make access control decisions [Hor97]. This has led to the development of security policy models that locate the control in the leaves rather than in the root of the access control structure [And96]
- An attempt by the UK government's Department of Health to introduce a conventional X.509 certificate structure [Zer96] was opposed by the medical profession. There was not just an ethical objection to the escrow features of the proposed key management protocol [JMW96]; it had also become clear that a central trusted third party could not cope with a health service whose approximately one million employees are

spread over 12,000 separate organisations.

- In addition to the engineering costs of centralisation, there was also the principle that electronic trust structures should mirror those in existing professional practice. This principle was first enunciated by Alexander Rossnagel, a German lawyer, who feared that certification authorities would deprive notaries of their income [Ros95]; it has since been adopted by the UK government and the medical profession as one of the principles by which disputes over electronic privacy issues are to be resolved [NHSE96].
- Governments attempted to solve security problems in Germany and Austria by introducing smartcards as access tokens for both patients and healthcare professionals. Despite careful government studies of the likely social consequences [BSI95], it has turned out that these cards have had a centralising effect that many professionals find intolerable [HW97].

The previous work that directly concerns us is Wax [Wax97a]. This is a proprietary hypertext system used for medical publishing; its goal is the secure and timely electronic distribution of information used to support clinical practice, such as treatment protocols and drug formularies. It will also be used for government circulars ranging from purely administrative information such as advice on coping with the Y2K bug to notices of newly discovered adverse drug reactions; and for local information such as hospital waiting lists.

Wax is already used in several health trusts in the UK for providing a mixture of trust-specific and general information. It is also used in the US for delivery of medical knowledge relating to HIV and AIDS by Intelligent Medical Objects, Inc. (Northbrook, Illinois, USA). There are clear safety and medico-legal reasons why the authenticity, integrity and timeliness of the information it distributes should be protected, and a project was undertaken during 1996-8 to design and implement this. That project is described in [Wax97a], but we will describe it here briefly so this article is self-contained.

2 Wax

The information in Wax is structured hierarchically with the levels being a shelf (owned by a publisher) containing books (each owned by an editor) consisting of chapters (each owned by an author). Thus if a primary care physician wants the latest advice on the conditions under which a patient with gout should be referred to hospital, he will draw from the Wax shelf the book on rheumatology and consult the chapter on gout. This chapter will have been written by a leading specialist and will be updated as necessary (typically every few years); the editor's job is quality control, principally choosing the experts and ensuring proper peer review.

As a solution was sought rapidly, an initial attempt at protection using digital signatures was undertaken using materials ready to hand – SHA, RSA, and X.509 [X509]. This decision was influenced by the fact that RSA with exponent 3 has just been accepted as the European standard for healthcare signatures. The X.509 hierarchy was founded on a Wax-root key, whose public component is embedded in the Wax browser software; Wax-root signatures certify keys of medical publishers (the Wax-centre for treatment protocols, the British Medical Journal, the Department of Health, individual hospital trusts, etc.) and the publishers in turn certify the keys of editors and authors.

As we did not know the optimum granularity of the signed objects, and had an operational requirement to open already cached books quickly, we also implemented a secondary protection mechanism whereby the book index contained (invisibly to the human reader) the SHA hashes of each chapter, and each shelf catalogue similarly contained hashes of book indexes. Thus a given book can be verified by means of its editor's signature, and also by reference to the publisher's catalogue. There is also considerable machinery to deal with trusted distribution of the

Wax software, trusted updating of local catalogues, and trusted collection of public keys from authors, none of which concern us here.

3 The first lessons

The principal lesson that we learned from this exercise was that the X.509 mechanisms are not really suitable for publishing. This realisation started to dawn when we had to decide on the longevity of publisher's keys. Assuming that users' keys would last three years, why not make the publishers' last five? But what would happen once a publisher's key was more than two years old, and thus unable to issue a certificate of three years' duration for an author? Would he have to refresh it, or acquire another one?

Many further complexities arose. For example, what does revocation mean in the context of book publishing? If an author fails to pay his annual fee to the local CA, will all his books magically vanish from all library shelves? What if a lawsuit is then brought in which a party relies on one of them? And what if revocation mechanisms were used maliciously as an instrument of censorship?

'Planned obsolescence' may make sense in software publishing, and in the banking world it is quite natural to use X.509 certificates in SET where both public and private keys have a lifetime of two years, as this simply replicates the existing trust structure of magstripe credit cards. However this approach is not appropriate in publishing, where objects are long-lived. Book copyright in the EU countries is now for a period of 70 years after the author's death.

The conclusion to which we were unexpectedly driven by the Wax project was that our secondary trust mechanism – namely, a tree of hashes in which chapters are hashed into a book and books into a catalogue – should in fact be the primary mechanism, while the X.509 signature mechanisms, which we had anticipated would provide the primary protection, are relegated to a number of secondary and specialist roles. The basic functionality can be seen in the following diagram (Fig. 1).

The question that we were naturally led to ask was whether catalogue-based trust had other natural applications than medical publishing, and what extensions of it might be appropriate. Our conclusion is that it gives a much better solution to some problems currently tackled using public-key cryptography, ranging from assuring the authorship of applets through enabling web authors to protect themselves from libel actions; in general, we can adopt the Wax mechanisms to provide a simple and robust set of mechanisms to authenticate the content of world wide web and other hypermedia systems, which fits well with the actual trust model that people have for published content.

We will now describe a set of proposed extensions to HTML that explain what we have in mind and make clear what can be achieved with it.

4 The Eternal Resource Locator

Trust, in the electronic world, is based on binding real-world assurances and/or relations to their electronic representation. This is expensive, and so in order for the trust transfer mechanisms (such as electronic signatures) to give maximum value, one should perform such bindings infrequently (but well). This is true for establishing a root of trust (e.g. top level Certification Authorities) and also for lower level entities. For example [CM97], it is a bad idea to bind keys and access rights to principals like this:

$$\text{key} \rightarrow \text{principal} \leftarrow \text{capability}$$

as this involves two bindings between the real world and cyberspace. We should rather build systems like this:

$$\text{principal} \leftarrow \text{key} \leftarrow \text{capability}$$

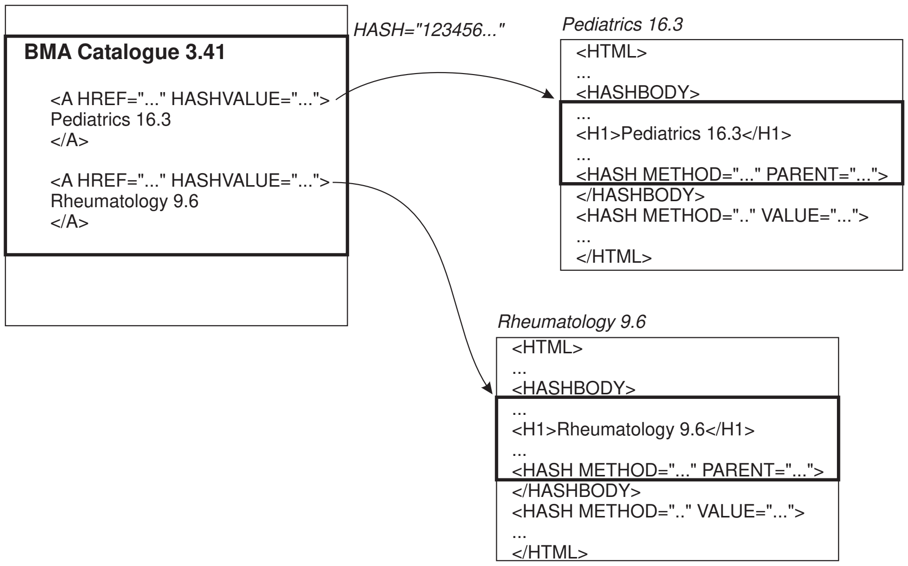

Figure 1: Publishing medical information. The publisher issues a catalogue every few months which lists all titles published, together with their hash values. The hash of the catalogue has to be distributed a trusted way, by being published in a paper journal, and signed using a long-term key.

Thus, when designing trust structures in general, we will try to have a small number of root keys or other authenticator values that can be made well known by out-of-band mechanisms, and derive the rest of the structure directly from these. The cleaner the structure, the better for a number of reasons, including both cost and robustness.

With trust based on hash trees, the root is quite simply the root of the tree; in the case of Wax, the hash of the current Wax catalogue. This can be assured by a variety of means (currently signature with the Wax-centre key and publication in the British Medical Journal). However, once we have gone to the trouble of certifying this root, we want all the pages in the publishing hierarchy to be checkable from it. We will now describe how this can be done using a simple extension that does not upset existing browsers, yet can be implemented either as a suitable applet or as part of a proxy service such as a hospital firewall. (The former is preferable as it is easier to implement 'untrusted' highlights, e.g. when a Wax chapter is retrieved as the result of a keyword search.)

4.1 Basic (static) mechanism

It would be clumsy to insist on the signature of whole web pages, so instead we propose to use HTML elements [HTML] to define the borders of the hashed section of the document as well as other features of the hashing mechanism:

- The `HASHBODY` element denotes the hashed section of an HTML document. All the text and HTML code in this section will be hashed with various hashing algorithm specified in the `HASH` element.
- The `HASH` element is an extensible container for use in identifying hash document information. It has three main functions: define the hashing algorithm used, store the corresponding hash value and link the element to a parent. The `HASH` element should be used both inside the `HASHBODY` section and outside; the purpose is to bind the protected section to its hash and its parent.

  There can be as many `HASH` elements as hashing algorithms. Attributes of the `HASH` element:

  - `METHOD` specifies the hashing algorithm. A number of algorithms may be used in parallel in order to give reassurance against cryptanalytic progress;
  - `VALUE` specifies the value of the hash;
  - `PARENT` provides a pointer to another HTML document, called parent and specified by its URL. This enables a browser that wishes to check the page's integrity to follow the hash chain to a suitable root. The name of the root may be given for performance reasons. If there is no parent (i.e. the document is a root) the attribute should not be omitted but instead should be set to NO.
  - The `URL` attribute optionally specifies where the page normally lives, and can provide basic protection against attacks involving the copying of pages to false hosts. Care is needed not to get entangled with HTTP server's interpretation of URLs like `http://www.foo.com/` and `http://www.foo.com/index.html`.

- We also propose to add new attributes to the existing `A` element. This will allow to define the hash of the resource specified by the `HREF` attribute. It is assumed that the linked resource is different from the current resource as the hashed body cannot include the hash value itself. Obviously, these can be two different fragments of the same web page.

  `HASHMETHOD`, `HASHVALUE` and `HASHPARENT` have the same meaning as the `METHOD`, `VALUE` and `PARENT` attributes of the `HASH` element.

Simplified, the way to protect part of the web page will look like:

```
<HTML>
...
<HASHBODY>
...
The examination results for the second MB
degree examination are as follows:
...
...
<HASH
URL="http://www.med.abc.ac.uk/examresults"
METHOD="SHA-1"
PARENT="http://www.cert.bma.org.uk">
<HASH
URL="http://www.med.abc.ac.uk/examresults"
METHOD="TIGER"
PARENT="http://www.cert.med.ac.uk">

</HASHBODY>

<HASH METHOD="SHA-1" VALUE="12345678..."
PARENT="http://www.cert.bma.org.uk">
<HASH METHOD="TIGER" VALUE="987654321..."
PARENT="http://www.cert.med.ac.uk">
...
</HTML>
```

One of the URLs that refer to this page might look something like:

```
...see <A
HREF="http://www.med.abc.ac.uk/examresults"
HASHMETHOD="TIGER" HASHVALUE="987654321..."
HASHPARENT="http://www.cert.med.ac.uk">here
</A>for the list of candidates who have
satisfied the requirements for the degrees
of MB and BS.
```

Checking a hash involves computing the hash value on all the bytes of an HTML document between the hash-input border tags and comparing the HTML document's URL against the value specified within the hash-input. This value is then verified against the value held in the reference in the parent document.

We call this URL-with-hash combination an ERL or 'eternal resource locator' as it makes static objects unique for ever. Dynamic objects are slightly more complex.

4.2 Dynamic pages

If we used hash functions alone, then this would limit us to material that was available and known when the last issue of the catalogue was published. Almost all published medical information is of this nature; it changes relatively slowly owing to safety protocols that insist on thorough peer review and validation. However, there is a demand for a small number of dynamic pages in the system holding 'hot' news such as recently advised drug side effects and other safety notices, or operational data such as test results. What we do not want to do is say something like 'for recent notices on test results X, look at URL Y for a message signed by a key certified to belong to Z' as this would suddenly involve reliance on a second root.

The effect would be that the referenced information was no longer part of the same trust structure, introducing complexity and making liability potentially uncertain. We therefore want to say something like 'for recent notices on test results X, look at URL Y for a message signed by a key whose hash is Z'. We will now describe the details.

The owner of the dynamic page creates a signature keypair and embeds the public key as follows:

```
...see <A
HREF="http://www.med.abc.ac.uk/bloodtest"
HASHMETHOD="TIGER" HASHVALUE="987654321..."
HASHPARENT="http://www.cert.med.ac.uk">here
</A> for today's blood test results for the
Fisher medical practice...
```

which refers to this:

```
...
<HASHBODY>
...
The blood test results for the Fisher practice
on 22/8/98 are as follows:
...
<HASH
URL="http://www.med.abc.ac.uk/bloodtest"
METHOD="SHA-1"
PUBLICKEYVALUE="ABCDEF01234.......89ABC"
ALGORITHM="RSAE3"
PARENT="http://www.cert.bma.org.uk">
<HASH
URL="http://www.med.abc.ac.uk/bloodtest"
METHOD="TIGER"
PUBLICKEYVALUE="ABCDEF01234.......89ABC"
ALGORITHM="RSAE3"
PARENT="http://www.cert.med.ac.uk">
</HASHBODY>
<HASH METHOD="SHA-1" VALUE="12345678..."
PARENT="http://www.cert.bma.org.uk"
SIGNATURE-VALUE="FEDCBA987..........76543"
ALGORITHM="RSAE3">
...
```

The reason that the public key's presence is not made clear in the parent page is to preserve bandwidth (keys are relatively large) and because we could find no reason why someone when clicking on a link should know in advance whether it is statically or dynamically protected. It also makes the implementation simpler.

Note that although we are using public key cryptography, we have no need of an X.509 certification mechanism. All the trust links created by the public keys are local and transient. So there is no need for long-term secrets; everything gets suddenly much simpler, more manageable and more exportable.

5 Other applications

Ignoring dynamic links for the moment, the trust structure naturally supported by ERLs has an interesting and, from any publisher's point of view, highly desirable property: that you cleanly distinguish material of which you approve and in which you expect your readers to place some reliance. This may seem trite but is a growing concern, as in the laws of many countries a defamation suit may be brought against anyone involved in the distribution of a contested statement and not merely the author.

In the UK it is normal for libel litigants to sue and attempt to enjoin the distributors of newspapers and magazines with which they have taken issue; recently the Nottinghamshire County Council issued lawyers' letters and injunctions against a number of people who had links on their home pages to leaked copies of a report on satanic child abuse that the council considered to be its copyright [Notts].

So putting a link on one's home page can be dangerous; the controller of the referenced page might introduce controversial material and one could be sued. The implications in medicine include, for example, that a hospital which carelessly referenced a drug company's information page could find its standing in a negligence case substantially altered; equally serious consequences could follow elsewhere.

So the general use of ERLs rather than URLs would often be prudent practice, as the failure of a followed link to authenticate will indicate that it has been changed since the author of the link last consulted it, and he can thus in no way be held liable for its contents.

Other applications will typically arise where a publisher owes some particular duty of care, and we suggest some examples below.

5.1 Public keys with multiple accreditors

It is quite common to assign a person a role whose performance depends on using a role key. However, we can have multiple parties having to approve assignment of such role. This is common in banking, where transactions over a certain amount typically have to be approved by more than one officer, but may be delegated on a limited basis. At present, special key management standards are being developed for banking by ANSI, as multiple signatures are not supported by X.509.

A similar problem arises in medicine, where the signing key used by (say) a doctor on a six month assignment in a hospital would not wish to use his long term personal signing key (hospital systems are often mutually incompatible and quite insecure) but would instead use a key that was signed both by his own personal long term key (in an off-line operation) and the hospital. This dual signature signifies that both the doctor and the hospital accept their joint liability for malpractice suits; it should also be possible for either of them to revoke the key when the relationship is terminated. This multiple revocation requirement is quite a complicated problem if one tries to implement everything as extensions of X.509.

Neither are medicine or banking the only applications in which dual control is required. Almost every substantial organisation has its own rules and procedures for managing dual control. Sometimes these rules may be hidden, as with military intelligence organisations who do not wish to reveal which officers have actual power; at other times, concealment is forced, as with the system of EU grants under which each grant receiving organisation (such as a university) must designate one 'official signatory' and the European Commission refuses to take any interest in the procedures that may lie behind this person's use of his official signature. How can we then support realistic trust models using either catalogue based or public key based trust mechanisms?

In the old days of paper-based banking systems, the custom was for each bank to print several hundred copies of a 'signature book', which contained the specimen signatures of its managers together with a set of rules defining, for example, which combinations of signatures were required on a letter of credit over \$10m. These books were then posted to its correspondent banks.

Our catalogue based trust mechanisms can provide an electronic implementation. In its simplest form, the company authenticates at regular intervals a set of public keys suitable for appropriate purposes and makes them available via web pages bound to the relevant trust trees.

This can even be done if need be in real time; we are experimenting with a mechanism whereby flexible links can be created on request to authenticate a key for a particular purpose. The example in the following diagram (Fig. 2) is where a company lawyer wishes to create a one-time key to conclude a property transaction that involves both internal certification (from his superior officer and the company's CEO) and an external land and property agency.

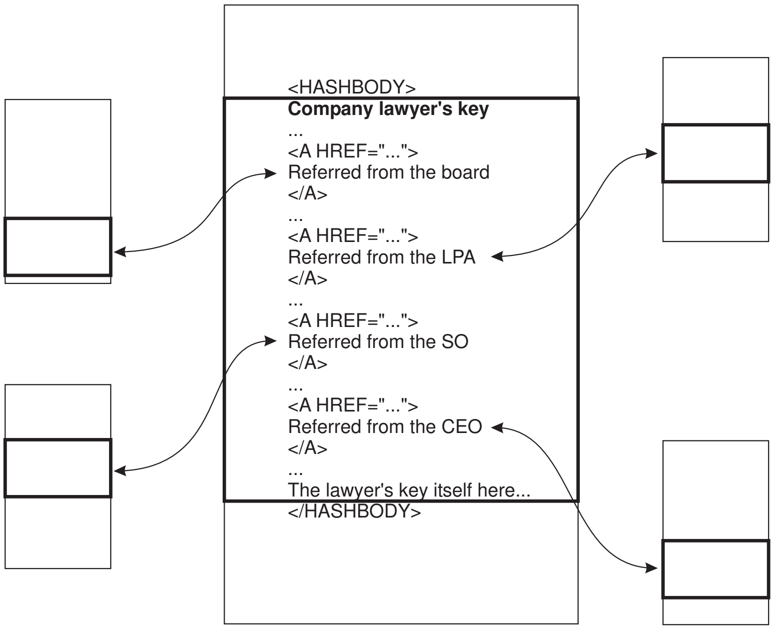

Figure 2: Publishing key information. Multiple accreditors are referring to the key that is valid if and only if all self-contained required links exist and the key-page hash value has not changed.

The effect of this mechanism is that CA functions can be performed on a one-off basis by various people and organisations as they are needed – a flexibility that is still critically lacking in X.509.

5.2 Timestamps

Time stamping services such as Surety's [HS91] are another example of a hash tree. In this case there is a mechanism for recomputing the tree and reliably publishing the hash every second, thus allowing rapid generation of an existence proof for a document.

We hope that formats for the inclusion of timestamps and other such evidence within the ERL structure and within HTML generally can be developed that is acceptable and useful to all parties.

5.3 Other considerations

The ERL idea also brings back the 'natural' concept of trust in a broader sense — it highlights the point of who trusts the certified entity, not the current notion of whom the certified entity trusts (or is forced to 'trust').

The ERL system requires reliable support of computing and checking hash values, in some cases supported by time-stamping services. It is also desirable to have the supportive mechanisms embedded within the existing computer environment (hardware, OS, Web browser or so).

We believe that our system is easily applicable on current Web platforms. The Web browser or OS should include features to compute the hash value of an arbitrary input (software distribution file, etc.) and perform a check on the hash of a loaded HTML document. It would be advisable to support several different hashing algorithms to avoid any future problem or failure. Also, hashes should be stored within the bookmark file of the browser, where they could provide some level of document change control exploitable for cache management, triggering warning mechanisms where relevant, etc.

6 The current version of Wax

The success of Wax led to interest from other countries. A US software company, Intelligent Medical Objects, Inc. (Northbrook, Illinois) decided to adopt the system as its preferred method for delivery of medical knowledge relating to HIV and AIDS. This would involve distribution of the browser software to over 300,000 physicians and other carers in the USA, and led immediately to a dilemma.

The owner of the RSA patent, RSA Data Security Inc., insists on a royalty that is a function of the sale price of software incorporating its technology, and which in the case of software distributed for free has a minimum value of \$5.00. The Wax project having been funded by charitable money, research grants and volunteer labour, was not in a position to pay \$1.5m as the price of entry to the USA.

This compelled the Wax project to revisit the cryptography issue. Another team was put together and we took a look at the design and trust issues. We found that we could achieve the same goals as before, and even more simply, by using one-time signatures instead of RSA. Necessity had truly become the mother of invention.

6.1 Application of one-time Signatures

In this subsection we present our solution based on one-time signatures. First we will recall briefly the ideas behind one-time signatures and then detail our implementation.

6.1.1 One-time signature basics

One-time signature scheme was first introduced by Lamport [DH76, Lam79] and more efficient schemes have been proposed since then [Mer87, Mer89].

We will assume here that the hash function h produces l bits and the message digests to be signed are n-bit long. The first step involved is the creation of a key-pair which will be used to sign a file only once; for this purpose, two arrays $\mathbf{X}$ and $\mathbf{Y}$ are generated. The first one contains $N = n + \lceil \log_2 n \rceil$ truly random l-bit-numbers $x_1 \cdots x_N$, and the second contains the hash values of these numbers, that is, $y_i = h(x_i)$. By definition the public key is: $K = h(\mathbf{Y})$.

The second step is to compute the signature of a file f whose hash is noted H_f. The signature of f is simply an array $\mathbf{S}$ whose N components are:

$$\begin{cases} s_i = (1-h_i)x_i + h_i y_i & \text{if } 1 \leq i \leq n, \\ s_i = (1-c_i)x_i + c_i y_i & \text{if } n < i \leq N, \end{cases}$$

where the h_i's are the binary digits of H_f and the c_i's the binary digits of a checksum $c = \sum_{i=1}^{l} h_i$. This checksum prevents attacks in which an opponent could produce a file f' such that all the '1' in H'_f are also in H_f but some '0' in H_f have been replaced by '1'. Once the signature is generated, the private key $\mathbf{X}$ should be destroyed.

Given f, $\mathbf{S}$ and K, verifying the signature implies: compute H_f, c, construct an array $\mathbf{Y}$ such that:

$$\begin{cases} y_i = (1-h_i)h(s_i) + h_i s_i & \text{if } 1 \leq i \leq n, \\ y_i = (1-c_i)h(s_i) + c_i s_i & \text{if } n < i \leq N, \end{cases}$$

and check that $h(\mathbf{Y}) = K$.

6.2 The current implementation

In the current version of Wax we apply the above one-time signature at the catalogue level. Each catalogue contains the hashes of all relevant books which can then be simply authenticated via the catalogue. We also link our catalogues together. We include in each signed catalogue seven public keys with which further editions of the catalogue, and other material from the same publisher, can be authenticated. This means that the chain of trust is broken if a user skips too many updates; in that case, he has to verify the public key of the new update using the same out-of-band mechanisms employed when the system was initially loaded and which we describe below.

For the initialisation of the chain of trust the Wax-centre generates eight key pairs $(\mathbf{X}_1, K_1) \ldots (\mathbf{X}_8, K_8)$. The first catalogue (C_1) contains the hashes of its associated files $f_{1,j}$ and the next public keys: $C_1 = \{h(f_{1,j})\}, K_2, \ldots K_8$. It is signed using the first private key: $\mathbf{S}_1 = s(C_1, \mathbf{X}_1)$. As mentioned above $\mathbf{X}_1$ is destroyed but the remaining $\mathbf{X}_i$'s are kept for the following distributions.

For the ith delivery a new key pair $(\mathbf{X}_{i+7}, K_{i+7})$ is generated and a new catalogue prepared:

$$\left\{ \begin{array}{lcl} C_i & = & h(C_{i-1}), \{h(f_{i,j})\}, K_{i+1}, \ldots, K_{i+7} \\ \mathbf{S}_i & = & s(C_i, \mathbf{X}_i). \end{array} \right.$$

Including the hash of the previous catalogue(s) into the current one prevent from denial of content attacks. Again, $\mathbf{X}_i$, the private key used for signature, is destroyed.

In order to bootstrap the trust in the system, each user is required to verify the public key K_1 of this initial shipment. A number of channels are provided for this, which is tightly bound up with the problem of trusted distribution. Initial deployment is by means of a mass mailing of CDs (stuck to the cover of an appropriate medical journal); electronic distributions are also available with authentication provided by the available mechanisms (such as PGP signatures, published MD5 hashes in medical journals, and download using SSL from a 'secure' web site).

The version of Wax that used RSA and X.509 had some further mechanisms, that were involved with users registering public keys of their own to the system; the corresponding private keys were used to generate signatures on books generated locally (such as treatment protocols developed in an individual medical practice) and also to generate counter-signatures on catalogues which had been downloaded and verified (as an extra precaution against virus attacks and the like).

On reviewing this design we concluded that the local use of public key cryptography added little. A medical practice which is going to publish a locally developed treatment protocol will as a matter of basic safety submit it to a peer review process, and thus all publication either is intermediated or can easily be made so. As for virus attack, the use of local signatures really only adds a modest layer of 'security through obscurity' as a virus written after study of the Wax code could alter the local public key and, absent tamper resistant processors, there appears to be no way to stop this.

6.3 New books

In order to issue an update of one or more books, a publisher just has to create a new catalogue and include the hashes of the books in it, then generate a new keypair and include the public part. The catalogue is now signed and made available for download together with the new book.

A user can choose which books he wishes to download and update. He must first download a catalogue and verify its signature. Then he can start downloading all or some of the books in the catalogue. Once these books have been downloaded and their integrity checked, the local index of books is updated and a checksum retained locally using 3-DES encryption and a passphrase, signed using a new one-time key pair.

6.4 New publishers

New publishers can introduce themselves at any time by simply publishing a catalogue, making it available for download, and providing out-of-band mechanisms for verifying the initial public key. This, however, replicates the effort required for out-of-band verification.

A simpler solution, which we have implemented, is to designate a special publisher (known as 'Wax-Root') whose sole function is to introduce new publishers. Each Wax-Root catalogue assigns one (or exceptionally more than one) of the 7 public keys to the new publisher's first catalogue, and in this way we branch the authentication tree.

7 Conclusions

This work has provided a number of insights.

Firstly, the use of hash trees rather than certificate chains is appropriate for trust relationships that change slowly (X is an employee of company A) or not at all (book Y was published by company B). Public key certificates are less suited for such relationships, at least in their most common forms: the typical three year lifetime of a key in an X.509 certificate system is too short for such applications; while a private signing key with a 100-year lifetime would be hard to protect (indeed, even three years may be too long a period to protect a really valuable private key). Catalogue based trust is one way of escaping this difficult trade-off between the need for a long-lived public key and a short-lived private one.

Secondly, trust mechanisms built using hash trees are simple to implement, intuitive to use, cost little in performance terms, and need contain no export-controlled mechanisms such as asymmetric cryptography.

Thirdly, catalogue based trust has a very natural fit with the publishing industry's business model in a number of ways ranging from the need to be careful about libel to the fact that catalogues are used anyway. Publishing is no longer a matter of the manufacture and distribution of books and newspapers; much of electronic commerce is publishing in some sense or another, and even where the net is used to sell widgets its main function is the publication of price lists, product data, delivery schedules and other information that is most easily organised in the form of one or more catalogues.

Fourthly, catalogue based trust mechanisms can be used to compensate for the shortcomings of X.509-type systems, such as the failure to support multiple certification discussed above.

Fifthly, catalogue based trust is robust. Cross-links can be inserted easily, in that a given book might appear in its publisher's sales catalogue and also in its editor's CV. Thus the security failure of one or another of these documents will leave the reliability of the book in unchanged. Such resilience is harder to achieve using X.509.

These advantages have become apparent to us in the course of the Wax project. We have sketched how very simple extensions to HTML can make them available to the net generally. We invite the authors, owners and proponents of other protection mechanisms to come together and agree a syntax for dealing with such protection tags in a standard way. The goal is no less than 'trusted browsing' – and, as this work makes clear, the admirable work already done in this direction by protocols such as SSL and SET is only the first step. There are many more protection goals than the secure transfer of credit card numbers from customer to merchant, and we believe that ERLs will make a significant contribution.

We have also developed a mechanism based on one-time signatures to assure the authenticity and integrity of electronic books. Although our particular application was medical, and was driven initially by a requirement to avoid RSA Data Security's patent, many of the lessons learned are much more general. We believe that the mechanisms described here are suitable for any application in which we need to assure the authenticity of relatively stable digital objects over long time periods, such as cataloguing, notarisation and archiving; they are certainly much more suitable than current incarnations of X.509 with all their expiry date and other problems.

There were other, less tangible, benefits. Moving from an X.509 implementation to this one-time scheme was like a breath of fresh air. Almost all the complexity vanished – from ASN.1 and DER, through modular arithmetic, to all the tricks used to protect local signing keys from casual attack. It was found that signatures based on one-way functions could be explained simply to the medical personnel involved in the project, as well as to programmers with no background (or interest) in cryptography. This made progress several times faster than had been the case when the RSA version was implemented in late 1996. The consequences for user trust in the system should not be underestimated; neither should the reduced likelihood that a design or programming bug will be discovered and exploited in attacks.

There will be applications in which a mixture of the number-theoretic and hash-function-based approaches will remain attractive. A book on investment, for example, might have its trust based on the techniques described here, but contain embedded public keys based on number theory in order to authenticate online pages of current stock prices. The advantage of such a structure is that these public keys now become independent of X.509 or any other public certification hierarchy, which is highly desirable given the lack of robustness of such mechanisms and the political struggles to control them. Such flexible links from a catalogue-based trust structure to more volatile items could, we believe, accommodate most of the world of journal and magazine publishing within the overall structure described here.

One current thrust of our research lies in extending the Wax mechanisms from proprietary to open publishing systems; ERLs will lay the foundation for this. Other directions of research include the control of updates to cached documents; allowing a user to store the hash with bookmarks and to be informed of changes – either when subsequently loading the document or at update; and interactions with the considerable range of other protection primitives in the security literature (anonymous messaging, digital cash, micropayments, incremental integrity primitives, copyright marking mechanisms and so on).

8 Acknowledgements

We would like to thank Iain Buchan and Rudolf Hanka for presenting us with the original challenge of providing security features for the Wax system, and to Bruce Christianson and Bruno Crispo for discussions on trust bindings.

9 Availability

More information on Wax, as well as the Wax software itself, can be found at

`<http://www.medinfo.cam.ac.uk/wax/>`

References

[And96] ‘A Security Policy Model for Clinical Information Systems’, RJ Anderson, in *Proceedings of the 1996 IEEE Symposium on Security and Privacy* pp 30–43

[BSI95] *‘Chipkarten im Gesundheitswesen’*, Bundesamt für Sicherheit in der Informationstechnik, *Bundesanzeiger*, 4 May 1995

[CM97] ‘Binding Bit Patterns to real World Entities’, B Christianson, JA Malcolm, in *Security Protocols Workshop 97*, Springer LNCS v 1361 pp 105–113

[DH76] ‘New Directions in Cryptography’, W Diffie and M Hellman, *IEEE Trans. on Information Theory*, IT-22, November 1976, pp 644-654

[HS91] ‘How to Time-Stamp a Digital Document’, S Haber, WS Stornetta, in *Journal of Cryptology* v 3 no 2 (1991) pp 99–112

[HW97] ‘Daten aus der Psychotherapie – auch bei uns bald eine Ware?’ A von Heydwolff, T Wenzel, *Psychotherapie Forum* v 5 no 1 pp 17–25

[Hor97] ‘Computers can be compatible with confidentiality’ JS Horner, in *Journal of the Royal College of Physicians of London* v 31 no 3 (May/June 97) pp 310–312

[JMW96] 'A Proposed Architecture for Trusted Third Party Services', N Jefferies, C Mitchell, M Walker, in *Cryptography: Policy and Algorithms*, Springer LNCS v 1029 pp 98–104

[Lam79] 'Constructing digital signatures from one-way function', L Lamport, *Technical Report SRI-CSL-98*, SRI International, October 1979.

[Mer87] 'A Digital Signature Based on a Conventional Encryption Function', RC Merkle, *Proc. CRYPTO'87*, LNCS 293, Springer Verlag, 1987, pp 369-378

[Mer89] 'A Certified Digital Signature', RC Merkle, *Proc. CRYPTO'89*, LNCS 435, Springer Verlag, 1990, pp 218-238

[NHSE96] Press release, NHS Executive, 17th October 1996

[HTML] 'HTML 3.2 References Specification – W3C Recommendation', Dave Ragget, January, 1997

[Notts] *Standard citation suppressed for legal reasons*; it can be found easily as 'The JET Report' using Web search engines.

[RRBL96] 'SDSI – A Simple Distributed Security Infrastructure', RL Rivest, B Lampson, at `<http://theory.lcs.mit.edu/~cis/sdsi.html>`, v1.0 presented at USENIX 96 and CRYPTO 96, April 30, 1996

[Ros95] 'Institutionell-organisatorische Gestaltung informationstechnischer Sicherungsinfrostrukturen', A Roßnagel, in *Datenschutz und Datensicherheit* (5/95) pp 259–269

[SHA] 'Secure Hash Standard', National Institute of Standards and Technology, *NIST FIPS PUB 180*, U.S. Department of Commerce, May 1993

[Wax97a] 'Secure Books: Protecting the Distribution of Knowledge', R.J. Anderson, V. Matyáš Jr., F.A.P. Petitcolas, IE Buchan, R Hanka, in *Security Protocols Workshop 97*, Springer LNCS v 1361 pp 1–11

[Wax97b] 'Coherent Exchange of Healthcare Knowledge in Open Systems', IE Buchan, R Hanka, in *Studies in Health Technology and Informatics*, C.Pappas et al (Eds), Vol 43, Medical Informatics Europe 97, pp 821–824, IOS Press, 1997

[X509] 'Information technology – Open Systems Interconnection – The directory: Authentication framework', *ITU-T Recommendation X.509*, November 1993; (June 1997 E)

[Zer96] 'The use of encryption and related services with the NHSnet', Zergo Ltd., published as NHSE IMG document number E5254, April 1996