Using WebDAV for Improved Certificate Revocation and Publication

David W. Chadwick and Sean Anthony

Computing Laboratory, University of Kent, UK

Abstract. There are several problems associated with the current ways that certificates are published and revoked. This paper discusses these problems, and then proposes a solution based on the use of WebDAV, an enhancement to the HTTP protocol. The proposed solution provides instant certificate revocation, minimizes the processing costs of the certificate issuer and relying party, and eases the administrative burden of publishing certificates and certificate revocation lists (CRLs).

Keywords: revocation, CRLs, LDAP, HTPP, WebDAV.

1 Introduction

The most common and standardized way of publishing X.509 certificates is in corporate LDAP servers. Several technical problems with the use of LDAP directory servers have been widely documented [1], including: the use ;binary encodings, the lack of a distributed directory capability, and the inability to search for certificates containing specific fields. Other problems are less technical in nature and instead are operational, but they are nevertheless just as inhibiting to successful deployments. For example, most corporate firewalls do not allow the LDAP protocol to pass through them, therefore certificates or certificate revocation lists (CRLs) cannot be easily accessed by external organizations. Finally, we have the complexity of installing and managing LDAP servers, for example, loading the correct schema and setting the correct access rights, which although not insoluble, nevertheless cause frustration and inconvenience especially to small scale deployments with a lack of specialist staff. For these reasons we wanted to find an alternative mechanism to LDAP for publishing X.509 certificates and CRLs that would not suffer from these problems. We wanted a generic solution that would support both public key certificates (PKCs) and attribute certificates (ACs), and that most (ideally all) organizations would already be familiar with. We chose to use Apache servers and the HTTP protocol, since these are ubiquitous. But HTTP on its own is insufficient, since it does not provide a number of useful features, such as the ability to search for specific content. For this reason we investigated (and subsequently implemented) the WebDAV extensions to HTTP [2].

The most common way of revoking public key certificates is through the use of CRLs. However these suffer from a number of well documented operational problems such as the potential for denial of service attacks from lack of availability, or the

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 265–279, 2007.

consumption of too many resources due to their increasingly large size. We will address the whole issue of certificate revocation in the next section, and describe why we have adopted an alternative approach for revocation based on WebDAV.

The rest of this paper is structured as follows. Section 2 re-appraisers the whole issue of certificate revocation and proposes a different approach to addressing this issue. Section 3 describes the WebDAV extensions to HTTP and how they can be used for X.509 certificate and CRL storage and retrieval. Section 4 describes our implementation of WebDAV in our PERMIS authorization infrastructure, in order to store X.509 attribute certificates used for authorization. This mechanism can similarly be used by PKIs to store public key certificates and CRLs. Section 5 discusses our approach, compares it to other work, and concludes.

2 Reappraising Revocation

There are several different approaches that have been taken to the complex issue of revocation of certificates, and of informing remote relying parties when revocation has taken place. A relying party is any Internet based service that consumes the issued certificate (whether it be a PKC or an AC). The primary objective of revocation is to remove a certificate (and all its copies, if any) from circulation as quickly as possible, so that relying parties are no longer able to use it. If this is not possible, a secondary objective is to inform the relying parties that an existing certificate in circulation has been revoked and should not be used or trusted. The latter can be achieved by requiring either the relying parties to periodically check with the certificate issuer, or the certificate issuer to periodically notify the relying parties. Of these, requiring the relying parties to periodically check with the certificate issuer is preferable for two reasons. Firstly, it places the onus on the relying parties rather than on the issuer, since it is the relying parties who are taking the risk of using revoked certificates. Secondly, an issuer may not know who all the relying parties are, so will have difficulty contacting them all, but the relying parties will always know who the certificate issuer is.

The simplest approach to certificate revocation, that used by X.509 proxy public key certificates [6], the Virtual Organisation Membership Service's (VOMS) X.509 attribute certificates [8] and SAML attribute assertions [7] (which are to a first approximation simply XML encoding of attribute certificates), is to never revoke a certificate, and instead to issue short lived certificates that will expire after a short period of time. The certificates are thus effectively and automatically removed from circulation after this fixed period expires. The assumption is that it is unlikely that the certificates will ever need to be revoked immediately after they have been issued and before they have expired due to abuse, therefore the risk to the relying parties is small. Risk (or more precisely risk exposure) is the probability of occurrence multiplied by the loss if the event occurs. Because short lived certificates are only valid for a short period of time, the probability of occurrence is small. Of course, the loss or amount of damage that can be done in a short period of time can be huge, so short lived certificates are not always the best solution where the resulting loss can be high. Consequently SAML attribute assertions have the optional feature of containing a "one time use" field which means that the consuming service can only use the attribute assertion once to grant access, and then it should never be used again. This is

designed to minimise the loss. A similar standardised extension could easily be defined for short lived X.509 public key and attribute certificates, in order to flag them for one time use. But this is not a ubiquitous solution since short lived certificates are not appropriate for every situation, nor is one time use.

An advantage of short lived certificates is that they effectively remove a certificate from circulation after a short period of time, and consequently they mandate that users or service providers must frequently contact the certificate issuer in order to obtain new freshly minted certificates.

The main disadvantage of short lived certificates is knowing how long to issue them for. They should be valid for the maximum time that any user is likely to need them for, otherwise one of the later actions that a user performs may fail to be authenticated or authorised which could lead to a session being aborted and all the processing that has been done so far, being lost. This is a current well known problem with the use of X.509 proxy certificates in grid computing. On the other hand, the longer the certificates are valid, the greater their possibility of misuse without any direct way of withdrawing them from circulation. This has caused some researchers to suggest that proxy certificates should be revocable!

A second disadvantage of short lived certificates is that the bulk of the effort is placed on the issuer, who has to keep reissuing new short lived certificates. This could become a bottleneck to performance. A better solution should put the bulk of the processing effort onto the relying parties, since these are the ones who want to use the issued certificates and the ones who need to minimise their risks. Thus it seems appropriate that they should be burdened with more of the costs.

If the primary objective of removing a certificate from circulation cannot be easily achieved, then the secondary objective of notifying the relying parties when a certificate has been revoked can be achieved by issuing CRLs. A CRL is a digitally signed list of revoked certificates, usually signed by the same authority that issued the original certificates. Revocation lists are updated and issued periodically. X.509 CRLs contain their date and time of issue, and have an optional *next update* time which signifies the latest date by which the next CRL will be issued. Relying parties are urged to obtain the next issue of the revocation list from the issuer's repository before the next update time has expired, in order to keep as up to date as possible. If the next update time is not specified in the CRL, then the frequency of update has to be communicated by out of band means from the issuer to the relying parties. Alternatively, the latest CRL can be sent by the subject along with his certificate, to prove that his certificates has not been revoked. The use of CRLs is standardised in X.509 [4] and the use of LDAP for storing CRLs in RFC 2252 [9]. Revocation lists ensure that relying parties are eventually informed when a certificate has been revoked, no matter how many copies of the certificate there are in circulation, but revocation lists have several big disadvantages. Firstly there is always some delay between a user's certificate being revoked and the next issue of the revocation list appearing. This could be 24 hours or even longer, depending upon the frequency of issuance of the CRLs. Thus, in order to reduce risk to a minimum, a relying party would always need to delay authorising a user's request until it had obtained the latest CRL that was published *after* the user issued his service request, which of course is impractical for most scenarios. If the relying party relies on the current revocation list, then the risk from using a revoked certificate equates, on average, to half that of using

a short lived certificate, assuming the validity period of a short lived certificate is equal to the period between successively issued CRLs. This reduced risk comes at an increased processing cost for the relying party and the issuer.

CRLs can put a significant processing load on both the issuer and the relying party. CRLs have to be issued at least once ever time period, regardless of whether any certificates have been revoked or not during that period. In a large system the lists can get inordinately long containing many thousands of revoked certificates. These have to be re-issued every time period, distributed over the network and read in and processed by the relying parties. In Johnson and Johnson's PKI, their CRL was over 1MB large within a year of operation [10]. To alleviate this problem, the X.509 standard defines delta revocation lists, which publish only the changes between the last published list and the current one. But this increases the processing complexity of the client, and few systems appear to support this feature today.

An alternative approach to notifying relying parties is to use the online certificate status protocol (OCSP) [3]. Rather than a relying party periodically retrieving the latest revocation list from the issuer's repository, the OCSP allows a relying party to ask an OCSP responder in real time if a certificate is still valid or not. The response indicates if the certificate is good, or has been revoked, or its status is unknown. Since most OCSP responders base their service on the latest published CRLs, the revocation status information is no more current than if the relying party had consulted the latest revocation list itself, thus the risk to the relying party is not lessened. But what an OCSP responder does do is reduce the amount of processing that a relying party has to undertake in order to validate a user's certificate. This reduced cost to the relying parties is offset by the cost of setting up and running the OCSP service.

We can see that none of the above approaches to revocation is ideal. Certificates often have a naturally long validity period. For example, authentication certificates are typically issued and renewed annually, whilst some authorisation certificates might require an equally long duration e.g. project manager of a 2 year project. Long lived certificates have traditionally necessitated the use of CRLs but they have several disadvantages. Alternatively we could issue short lived session certificates throughout the duration of the natural validity period, for both authentication and authorisation purposes, without needing to issue CRLs as well, but then there is the inherent conflict between making the short lived certificates long enough for the biggest session and short enough to minimise the risk. Thus we propose a new conceptual model that we believe is superior to short lived certificates, CRLs and OCSP servers.

2.1 A New Model for Revocation

We believe that the optimum approach to certificate issuing and revocation should have the following features:

- A user's certificate should only need to be issued once (and not continually reissued as with short lived certificates) in order to minimise the effort of the issuer.
- The certificate should be valid for as long as the use case requires, which can be a long (measured in years) or short (measured in minutes) duration. Again, this minimises the effort of the certificate issuer (and the delegator, when attribute certificates are used to delegate authority).

- A certificate should be able to be used many times by many different relying parties, according to the user's wishes, without having to be reissued. Of course, the certificate will need to be validated by each relying party each time it is used. But this mirrors the situation today with our plastic credit cards and other similar types of certificate.
- A certificate should be capable of being revoked at any time, and the revocation should be instantaneous. Relying parties should be able to immediately learn about revocations thereby minimising their risks.

All the above features, including instant revocation, can be achieved in the following way. The issuer issues a certificate, giving it its natural validity period, and stores the certificate in a HTTP based repository which is under the control of the issuer. We choose HTTP since this protocol can penetrate firewalls and provide read access to external relying parties. Each certificate is given its own unique URL (called the certificate URL) which points to its location in the repository. This URL is stored in the certificate in a standard extension, in order to strongly bind the repository location to the certificate, so that relying parties know where to go to check if the certificate is still valid. In order to revoke a certificate, the issuer simply deletes the certificate from its repository. The absence of a certificate at its published URL indicates that it is no longer valid. As an added security measure, the issuer may also simultaneously issue a CRL of length 1 and store this at another unique URL (called the revocation URL). The revocation URL, if used, should also be inserted into the original certificate using the existing standard CRL distribution points extension. Relying parties are now able to instantaneously find out the current status of a certificate by contacting the issuer's repository, using either of the URLs embedded in the certificate.

The frequency and method by which a relying party contacts the issuer's repository is determined by its risk mitigation strategy and the optional presence of the revocation URL. The frequency can vary per application or per user request, and is set by the relying party as appropriate, and not by the issuer, which is putting the responsibility and risk where it belongs, with the relying party. In order to minimise risk, a relying party should contact the issuer's repository when a certificate is first validated, and then periodically during the life of the user's session according to its own risk assessment. If the relying party is operating in certificate pull mode, then it must contact the repository anyway at first use in order to pull the certificate, but if the relying party is operating in certificate push mode, contacting the repository is optional from a technical perspective. It must therefore be determined from a risk perspective.

We propose the following procedure for determining the revocation status of a certificate. The relying party periodically issues a HTTP or HTTPS GET command to the certificate URL. (As previously stated, this should be when the certificate is first validated, and then at risk determined intervals.) We will discuss the choice between HTTP and HTTPS later. If the HTTP status code 404 Not Found is returned, the relying party may assume that the certificate has been revoked, and permanently record this in its internal cache along with the time of the request. If the certificate is returned, a simple bitwise comparison of the initial validated certificate with the subsequently retrieved copy of the certificate is all that is needed by the relying party

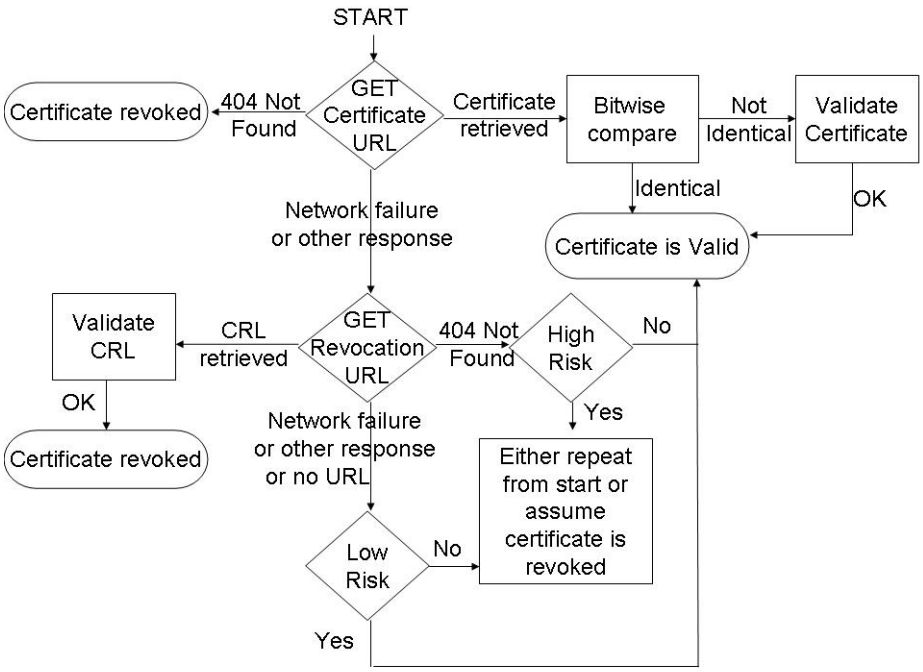

Fig. 1. The revocation procedure

to ensure that the certificate is still identical to the one originally validated. Certificate signature verification is therefore not needed for revocation status checking. The relying party may optionally cache the valid certificate and its time of last retrieval. If the certificate has been updated in the repository during the user's session, then the retrieved certificate will fail the bitwise comparison and will need to be validated again, but this should be a relatively rare occurrence. This procedure is designed to minimise the processing effort of the issuers and the relying parties, whilst maximising the freshness of the revocation information. Issuers do not need to continually mint new certificates or CRLs, and relying parties do not need to process potentially large revocation lists or perform expensive cryptographic operations for the vast majority of their revocation checks.

If on the rare occasion a client is unable to contact the certificate URL and retrieve either a certificate or a Not Found response, it may attempt to contact the revocation URL, providing there is one is in the certificate. If the HTTP status code 404 Not Found is returned from the revocation URL, the relying party may assume that the certificate is still valid (except in high risk cases, where an attacker may be blocking access to the certificate URL and spoofing the revocation URL), and optionally cache this result along with the time of the request. If the CRL is returned instead of Not Found, the signature is validated and the relying party caches the result permanently to ensure the certificate cannot be used again and no further retrieves need be made.

Intermediate caching of the CRL is supported and encouraged, so that if a certificate has been revoked and the CRL successfully retrieved, intermediate web servers can cache the CRL to speed up subsequent queries. Finally, if the relying party is unable to make a connection to the revocation URL, or one does not exist, then the relying party can check its cache to see if a previous request to either URL has succeeded or not. If neither URLs are available, the relying party should use its local risk assessment procedure to decide what to do when there are network problems. For example, if the transaction is low risk, it may decide to treat the certificate as valid. Alternatively it may decide to try contacting the URLs again, or alternatively to treat the certificate as revoked. The flow chart in figure 1 summarises this procedure.

Clients may use either HTTPS or HTTP depending upon their and the issuer's security requirements. HTTP presents a number of security weaknesses compared to HTTPS. Firstly HTTP provides public access to the certificate, which may violate the privacy of the certificate subject. (There is no equivalent privacy leakage for a CRL.) Furthermore intermediate Web servers may cache copies of frequently accessed web pages to improve performance, but this would negate the proposed revocation service. To counteract this, the issuer's Web server must use the no-cache cache-response-directive [15] in the HTTP response of successful certificate requests and Not Found CRL requests, to prevent intermediate servers from caching these responses. This will ensure that all subsequent queries are directed to the authoritative source of the information and that stale cached responses are not received. Finally HTTP is susceptible to redirection, substitution and man in the middle attacks. Consequently, if the certificates are not meant to be publicly available or stronger security is required, then secure access should be provided using HTTP with TLS [5]. This will stop network redirection, substitution attacks and intermediate caching. TLS can also provide confidentiality of the retrieved certificates during transfer, in cases where privacy protection of sensitive certificates is required by the issuer. TLS can also provide strong client side authentication, which will allow access controls to be placed on the WebDAV repository, further protecting the privacy of the subjects' certificates. The privacy of CRLs is less important, and it enhances security if more copies of these are publicly available.

3 The WebDAV Protocol and Its Use with X.509

WebDAV [2] is an Internet RFC that specifies extensions to the HTTP/1.1 protocol so that web content can be managed remotely. WebDAV provides users with the ability to create, remove and query information about web pages, including their contents and properties, such as their creation dates, expiry dates, authors etc. In the context of X.509, a web page will be a single X.509 certificate (either public key or attribute) or a CRL containing a single entry, and their properties can be any fields of the certificate or CRL. WebDAV also provides the ability to create sets of related web pages, called collections, and to retrieve hierarchical membership listings of them. In the context of X.509, a certificate subject can represent a collection, and his/her

certificates can be the collection membership listing. The set of CRLs issued by an issuer can also be a collection membership listing. WebDAV is widely supported, several open source implementations are available including one for Apache, and there is an active community working with it (see http://www.webdav.org/).

WebDAV resources are named by URLs, where the hierarchical names are delimited with the "/" character. The name of a collection ends with /. If we model our X.509 certificate store in the same way as an LDAP directory tree, and name it using the subject DNs to represent collections, this provides us with the ability to retrieve a listing of all the certificates that are owned by a single subject. For example, a public key certificate belonging to the subject whose Distinguished Name is c=gb, o=University of Kent, cn=David Chadwick, might be named in a WebDAV repository with the URL:

https://server.dns.name/c=gb/o=University%20of%20Kent/cn=David%20Chadwick/pkc=Verisign%20Class1.p7c

Note that the last component pkc=Verisign%20Class1.p7c is the unique name (in terms of the collection) given to the certificate by its issuer. We do not mandate any specific values here, but we recommend using the following file extensions: .p7c for public key certificates, .ace for attribute certificates and .crl for CRLs. A GET request to retrieve all the certificates of David Chadwick would use the URL of the collection, viz:

```
GET /c=GB/o=University%20of%20Kent/cn=David%20
Chadwick/ HTTP/1.1
Host: server.dns.name
```

We can similarly model a CRL store as a collection under its issuer, using the collection name cn=CRLs/, and name each CRL that is issued with the serial number of the certificate that it revokes. This provides us with the ability to retrieve a listing of all the CRLs that have been issued by a single issuer. For example, if David Chadwick is an attribute authority who delegates an attribute certificate with serial number 1234456 to another person in his organization, and then subsequently revokes the AC, the CRL would be located at:

http://server.dns.name/c=gb/o=University%20of%20Kent/cn=David%20Chadwick/cn=CRLs/serialNumber=1234456.crl

A GET request to retrieve all the CRLs issued by David Chadwick would use the URL of the collection, viz:

```
GET /c=GB/o=University%20of%20Kent/cn=David%20
Chadwick/cn=CRLs/ HTTP/1.1
Host: server.dns.name
```

In order to create a new collection, WebDAV specifies the MKCOL method. The difference between this method and HTTP PUT or POST, is that the latter are allowed to overwrite existing content at the specified URL, whereas MKCOL will fail if there is any existing content at the specified URL. In the context of X.509, this ensures that a certificate issuer cannot unwittingly overwrite existing certificates when creating a

new collection for a subject. This is an important concern when there are several certificate issuers for the same subject (either attribute certificate issuers and/or public key certificate issuers). It is important to ensure that no issuer deletes the certificates issued by another issuer.

In order to create a certificate or CRL or update an existing certificate in an existing collection, the PUT method is used. It is essential that every certificate and CRL has a unique name within a collection, so that updates can overwrite the same certificate and new certificates and CRLs cannot overwrite existing ones. The onus for creating the unique names is with the issuer. We have defined our own algorithms for ensuring this uniqueness is maintained in our implementation, see Section 4 below.

In order to revoke a certificate, the HTTP DELETE command is used by the issuer. This removes the certificate and its properties from the WebDAV server. Simultaneously with this, if CRLs are supported, the issuer should use the HTTP PUT method to create a new CRL containing the serial number of the certificate that has just been revoked. The revocationDate and thisUpdate fields of the CRL should be set to the current time, and the nextUpdate field should be set to sometime after the certificate was due to expire, thereby ensuring that the CRL never needs to be reissued or updated.

Document properties are specified in XML as name/value pairs. Property names must be globally unique and are specified using XML namespaces [11]. Property values should be human readable (in any appropriate character set). Properties can be flagged as live or dead, where live means that the server validates that the values are syntactically correct, and the server may actually set the values of some system known properties, whereas dead means that the client is responsible for validating the syntax and semantics of the property values. For X.509 use, we initially intended to use live properties, set by the certificate issuer, to represent fields of the certificate. We anticipated this would allow easy searching of the certificate store to find certificates with certain properties, for example, find the AC of David Chadwick that has a *manager* role value. The WebDAV protocol does support the PROPFIND method, in which the properties of a resource can be retrieved, but it not possible to specify which property value you require. Only the property types can be specified. Consequently, if we perform a PROPFIND for the "Role" property, then the web server will return an XML encoded message containing all the ACs that contained a property named Role along with their values. Clearly this is not a viable solution. Work on the WebDAV searching and locating capability (DASL) started in 1998, but the work was never completed and the IETF closed the DASL working group some years later. The latest version of the WebDAV Searching and Locating protocol is very recent [17] and several implementations are said to exist, but we were unable to find a usable one. Consequently we have left the search feature for future work. Instead we have implemented a browsing capability in our user agents which allows a user to tree walk through a certificate store and select the certificate that he is looking for. The browse capability is fully scalable, user friendly and meets all the requirements of our use cases. See section 4 and figure 3 for more details.

WebDAV also provides other features that we do not need for X.509 use, such as the ability to copy and move web documents between servers, and the ability to write lock the certificate store when an issuer is performing updates. Consequently, these wont be discussed further.

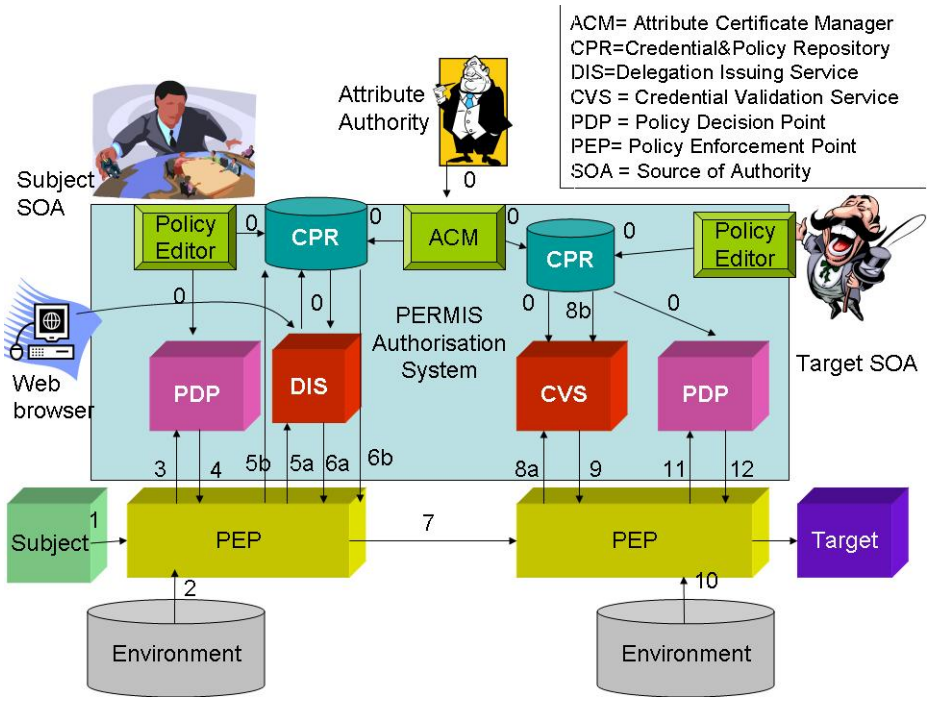

Fig. 2. The PERMIS Authorisation Infrastructure

4 Using WebDAV in PERMIS

PERMIS [12, 13] is an application independent privilege management infrastructure that comprises credential issuing and credential validation functionality as well as policy creation and policy decision making functionality. The components that are important to the current discussion are the Attribute Certificate Manager (ACM) and Delegation Issuing Service (DIS), which both issue X.509 role ACs to holders, and the Credential Validation Service (CVS) which validates the issued role ACs (see Figure 2). In addition, the Policy Editor creates XML policies to control the behaviour of the DIS, CVS and PDP, and each policy can be embedded as a policy attribute in an X.509 AC and digitally signed by its issuer. The policy AC can then be stored in the issuer's collection of ACs, along with his role ACs. Note that the only difference between a role AC and a policy AC is the content of the attribute that is embedded in the AC (although the holder and issuer of a policy AC always contains the same DN). In the original implementation of PERMIS, all the issued X.509 ACs were stored in LDAP directories, in the attributeCertificateAttribute of the entries of their holders. A major disadvantage of this, is that it is impossible to retrieve a single AC of a holder. Instead the entire set of ACs (role and policy ACs) held by a holder has to retrieved as a set. The latest implementation now has the ability to store the ACs in and retrieve them from WebDAV repositories, in which each AC is uniquely identified.

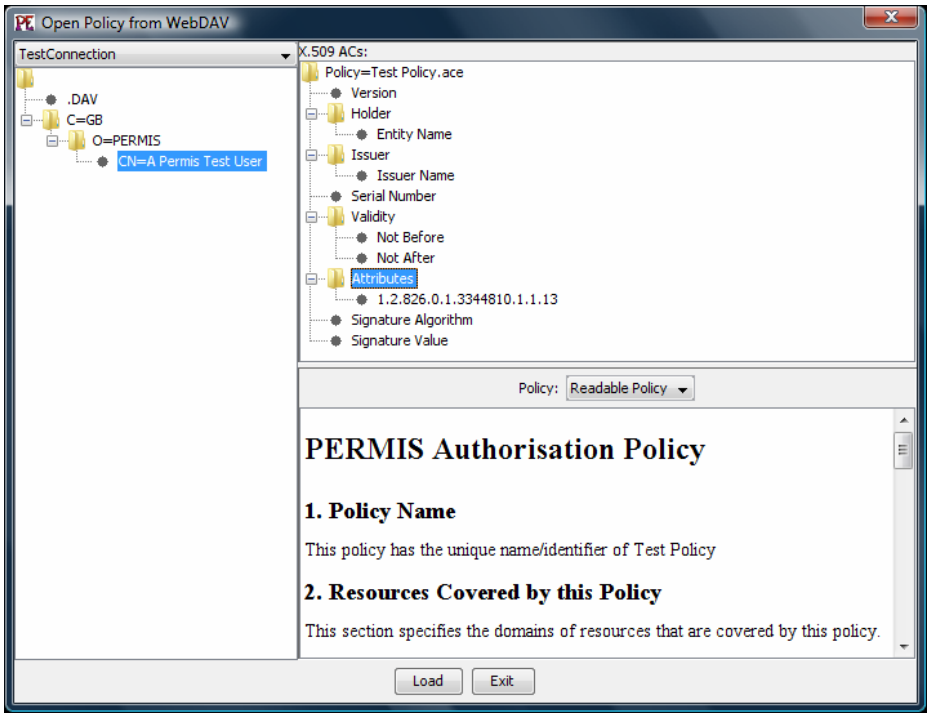

Fig. 3. Retrieving a Policy AC using WebDAV

4.1 Deriving Unique Names for Certificates and CRLs

Because policy and role ACs may be updated by their issuers it is important to have unique names for each of them. Furthermore, all role ACs must have their unique certificate URL and optional revocation URL embedded in extensions so that relying parties can retrieve the contents of either URL to check that the role AC has not been revoked. Rather than allowing the user to specify the names of the ACs or CRLs, we algorithmically create the unique names as follows:

- each AC has the file suffix .ace, whilst each CRL has the file suffix .crl
- the name of a role AC file is created from the contents of its first embedded attribute value plus the serial number of the certificate E.g. a role AC with the embedded attribute type PermisRole with attribute value Project Manager, and certificate serial number of 12345 would create the filename "PermisRole=Project Manager+SN=123456.ace". The serial number provides the uniqueness, whilst the attribute type and value provides user friendliness when the issuer wants to browse WebDAV and retrieve an AC for editing (see Figure 3). As an additional user friendly feature, the WebDAV AC browser displays the entire contents of all the attributes in the bottom window so that the user is sure that he is retrieving the correct role AC.
- the name of a policy AC file is created from the unique name of the embedded XML RBAC policy, which is an XML attribute of the RBAC Policy Element.

E.g. a policy with the name "AstroGridUsers" would produce the name "Policy=AstroGridUsers.ace" (see Figure 3). As an additional user friendly feature, the WebDAV policy browser displays the content of the XML policy attribute in either raw XML or natural language (see lower screen) so that the user is sure that he is retrieving the correct policy.

- the name of a CRL file is created from the serial number of the certificate that it revokes. E.g. a CRL that revokes a certificate with serial number 1234 would produce the filename "serialNumber=1234.crl".

4.2 Certificate Extensions

The optional revocation URL can be placed in the existing standard CRL distribution points extension. The certificate issuer should place the HTTP URL of the future CRL in the uniformResourceIdentifier component of the GeneralName of the DistributionPointName. Note that this URL will not exist until the certificate has been revoked, therefore it is important that the issuer has a deterministic algorithm for creating these URLs, such as the one given in section 4.1 above.

In order to place the certificate URL in an X.509 extension field, we define a new access method for the AuthorityInformationAccess (AIA) extension defined in RFC 3280 [14]. The AIA extension is designed to point to services of the issuer of the certificate in question. One of the standard uses of this extension is to point to the OCSP service provided by the issuer. Since our WebDAV service is replacing the OCSP service, it seems appropriate to use the AIA extension to point to our WebDAV service. We copy below the ASN.1 of the AIA extension, taken from [14] for the convenience of the reader:

```
AuthorityInfoAccessSyntax  ::=
      SEQUENCE SIZE (1..MAX) OF AccessDescription

AccessDescription  ::=  SEQUENCE {
      accessMethod        OBJECT IDENTIFIER,
      accessLocation      GeneralName  }
```

We now define our new accessMethod, webdav, as follows:

webdav OBJECT IDENTIFIER ::= { 1.2.826.0.1.3344810.10.2 }

When the AIA accessMethod is webdav, then the accessLocation must be a URL pointing to the WebDAV server where the certificate can be found. The URL must point to the exact location of the certificate in the server so that relying parties can download the certificate to compare it to the copy they hold. The absence of the certificate at the URL of this WebDAV server means that the certificate has been revoked.

The Object Identifier in the definition above is one that we have allocated ourselves. However, we propose to take this definition to the IETF PKIX group for standardisation, so that the OID can be replaced by one defined by the PKIX group.

5 Discussion and Conclusions

The OASIS SAML specification has the concept of an artifact that can be obtained from a remote server using an ArtifactResolve message to request the artifact and an ArtifactResponse message to return it [16]. The artifact could be a SAML Attribute Assertion, which is similar in concept to an X.509 attribute certificate, except that it is designed to be short lived and never revoked. The SAML artifact messages are carried over HTTP, therefore will pass transparently through firewalls in the same way as our WebDAV protocol. But there the similarity between the two schemes ends. There are fundamental conceptual differences between the artifact concept in SAML and the certificate publishing and revocation concepts in this paper. Firstly a SAML artifact can only be used once. The SAML specification states "The responder MUST enforce a one-time-use property on the artifact by ensuring that any subsequent request with the same artifact by any requester results in an empty response" [16]. Secondly SAML artifacts are meant to be short lived, quote "The artifact issuer SHOULD enforce the shortest practical time limit on the usability of an artifact, such that an acceptable window of time (but no more) exists for the artifact receiver to obtain the artifact and return it in an <ArtifactResolve> message to the issuer" [16]. In our design, certificates are assumed to be as long lived as required, and used as many times as needed by as many different recipients as the subject desires. We thus believe are system is more flexible and requires less processing resources on the part of both the issuer and relying party.

Our scheme has some similarities with the Netscape Navigator certificate revocation mechanism [18]. In the Netscape scheme, an X.509 extension **netscape-revocation-url** is used to refer to a web location where information about a certificate's status can be found. The actual URL that a relying party should use comprises this extension concatenated with the certificate's serial number (encoded as ASCII hexadecimal digits) e.g. https://www.certs-r-us.com/cgi-bin/check-rev.cgi?02a56c. The document that is retrieved from this URL contains a single ASCII digit, '1' if the certificate is not currently valid, and '0' if it is currently valid. The differences with our scheme are immediately obvious. The revocation URL document always exists, and its content is not digitally signed by the issuer. In comparison our certificate only exists whilst it is has not been revoked and it is a standard certificate digitally signed by the issuer. Optionally our CRL only exists if the certificate has been revoked, and it is a standard CRL containing a single entry signed by the issuer. Our scheme also optionally allows a relying party to find out all the certificates that have been revoked by a particular issuer, by retrieving the WebDAV CRL collection (cn=CRLs/) under the issuer's WebDAV entry.

The one security weakness in our scheme is that it is vulnerable to denial of service attacks, in that if the WebDAV server is not available, relying parties will not be able to tell if a certificate has been revoked or not. But other schemes such as OCSP servers and published CRLs are also equally vulnerable to DOS attacks, and so our scheme is no different in this respect. However, published CRLs do have one advantage in that an old CRL retrieved sometime in the past might still be available to the relying party, and this is better than having no CRL at all, since it does contain some revoked certificates. If this is seen to be a significant benefit, then our optional CRL publishing mechanism is equivalent to it, in that a CRL collection can be

downloaded at any time, just like a conventional CRL. The CRL collection can also be replicated and cached to improve availability. Other well known DOS protection methods, such as overcapacity and server clustering will have to be employed in order to be fully protected against DOS and DDOS attacks, but these are fairly standard techniques that are employed by DNS servers and commercial web sites such as Amazon. Consequently we do not believe that DOS attacks are any more of a significant security threat to our scheme than to existing ones.

The one performance weakness of our scheme is that the latency of certificate validation increases due to network overheads, as compared to that of traditional CRLs, but not to that of OCSP servers. This may be a critical factor to some relying parties such as central servers which process thousands of certificates per second. Central servers benefit from downloading traditional CRLs when they are not busy and storing the results in a local cache for fast lookups when they are validating certificates. The disadvantage of this approach is that the central server still has a vulnerability period between the date and time the latest CRL was published and now, during which all recently revoked certificates will not yet have been incorporated into the latest CRL. This provides an attack window for the holders of the recently revoked certificates. If on the other hand the certificate issuer supports our WebDAV certificate publishing scheme alongside its traditional CRL publishing (or our WebDAV single CRLs scheme) then the central server may check the current status of certificates. It can use the WebDAV GET operation for the certificates of high risk or high value transactions, whilst continuing to use its CRL cache for the certificates of lower risk transactions. In this way the latency penalty of WebDAV lookups is only incurred for a few certificate validations.

In contrast, low throughput relying parties which only process a certificate every few minutes, and where the subject base is very large (and hence so are the traditional CRLs) will benefit greatly from our WebDAV approach of contacting the certificate URL at the time of each certificate validation. This is not too dissimilar from contacting an OCSP server, in terms of processing overheads (HTTPS overheads vs. signed OCSP responses) and latency. The advantages of our WebDAV scheme are that HTTPS and web servers are more ubiquitous than OCSP servers, and where OCSP servers compute their responses on published CRLs and therefore are out of date, WebDAV responses are based on the latest up to date certificate status information.

Finally, comparing our single CRL per certificate scheme against traditional CRLs, we see that there is a trade off between the currency of the revocation information and the overhead of signature creation and validation. A traditional CRL only requires one signature creation per revocation period and one signature validation per relying party per period, whereas our scheme requires one signature creation per revoked certificate and one signature validation per relying party per revocation. The more certificates are revoked per revocation period, the more the processing overhead increases, but so does the risk. Consequently the increased processing overhead has to offset against the risk reduction of instant revocation, but this tradeoff can only be determined on a per-application basis.

To conclude, we have described a new way of publishing and revoking X.509 certificates based on the ubiquitous WebDAV protocol that has a number of distinct advantages over current schemes. We have implemented this in our PERMIS

privilege management infrastructure and performed initial user testing. It has recently been released as open source software along with the existing PERMIS source code, and we will soon expect to obtain operational experiences from users.

References

1. Chadwick, D.W.: Deficiencies in LDAP when used to support a Public Key Infrastructure. Communications of the ACM 46(3), 99–104 (2003)
2. Goland, Y., Whitehead, E., Faizi, A., Carter, S., Jensen, D.: HTTP Extensions for Distributed Authoring – WEBDAV. RFC 2518 (February 1999)
3. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol – OCSP, RFC 2560 (1999)
4. ITU-T. The Directory: Public-key and attribute certificate frameworks" ISO 9594-8 (2005) /ITU-T Rec. X.509 (2005)
5. Dierks, T., Allen, C.: The TLS Protocol Version 1.0, RFC 2246 (January 1999)
6. Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M.: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC3820 (June 2004)
7. OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard (15 March, 2005)
8. Alfieri, R., Cecchini, R., Ciaschini, V., Dell'Agnello, L., Frohner, A., Lorentey, K., Spataro, F.: From gridmap-file to VOMS: managing authorization in a Grid environment. Future Generation Computer Systems 21(4), 549–558 (2005)
9. Wahl, M., Coulbeck, A., Howes, T., Kille, S.: Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions. RFC 2252 (December 1997)
10. Guida, R., Stahl, R., Bunt, T., Secrest, G., Moorcones, J.: Deploying and using public key technology: lessons learned in real life. IEEE Security and Privacy 2(4), 67–71 (2004)
11. Bray, T., Hollander, D., Layman, A.: Namespaces in XML. World Wide Web Consortium Recommendation REC-xml-names-19900114. See http://www.w3.org/TR/1999/REC-xml-names-19990114/
12. Chadwick, D.W., Otenko, A., Ball, E.: Role-based access control with X.509 attribute certificates. IEEE Internet Computing, 62–69 (March-April, 2003)
13. Chadwick, D., Zhao, G., Otenko, S., Laborde, R., Su, L., Nguyen, T.A.: Building a Modular Authorization Infrastructure, UK All Hands Meeting, Nottingham (September 2006). Available from http://www.allhands.org.uk/2006/proceedings/papers/677.pdf
14. Housley, R., Ford, W., Polk, W., Solo, D.: Internet, X.: 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 3280 (April 2002)
15. Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (June 1999)
16. OASIS. Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard (15 March, 2005)
17. Reschke, J., et al.: Web Distributed Authoring and Versioning (WebDAV) SEARCH. <draft-reschke-webdav-search-11> (9 February, 2007)
18. Netscape Certificate Extensions, Navigator 3.0 Version. Available from http://wp.netscape.com/eng/security/cert-exts.html