Slovak University of Technology, Bratislava

Faculty of Electrical Engineering and Information Technology

Internet applications security

Written part of Ph.D. Exam

Ing. Radovan Semančík

November 2002

Supervisor: Doc.Ing. Margaréta Kotočová, CSc.

Doctoral Study Program: 25-21-9 Computer Systems

Table of Contents

1 Introduction

The Internet becomes an universal communication medium of this century. Internet links carry all the different kind of traffic: web pages, interactive sessions, business transactions, voice, video and many more. Most of the information carried by the Internet is private and needs an appropriate protection. It is apparent that a mechanism is needed to authenticate different Internet users and control their access to resources. Many authentication schemes and security infrastructure architectures were developed, but not all of them are suitable for Internet global environment. Service levels provided by different schemes vary in scalability, manageability and efficiency. Selected security methods are considered in this work and their suitability for Internet environment is evaluated. Special attention is paid to the schemes suitable for the emerging web services environment. Several digital identity systems are considered in this work, including Microsoft Passport and Liberty Alliance Project and their characteristics are evaluated. The Security Association Markup Language (SAML) is described as a firm base for the digital identity systems and web services security mechanisms.

The next section provides an overview of the relevant Internet technologies. The rest of this work is based on these technologies and provides an description of security mechanisms that are used to secure the Internet applications and services.

The section 3 covers the authentication and access control systems that are used on the Internet today. It is focused specifically on the X.509 PKI..

The section 4 describes digital identity systems and evaluates their characteristics. Special attention is paid to the Security Association Markup Language (SAML) and its use in the identity systems.

2 Overview

Internet applications are distributed applications built on the internet protocols and operate in the global Internet environment. Many different application architectures were used in this environment, but since the spread of WWW technologies these became the primary application platform of the Internet.

2.1 World Wide Web

The World Wide Web (WWW) technology emerged in early 90's and was designed for a hypertext management in the scientific community. However, it was soon discovered that the full potential of WWW was far beyond the plain hypertext functionality.

2.1.1 Static Web Content

First web pages were plain scientific hypertext documents, but as the acceptance of the WWW grew, more and more features were introduced. When the first graphical WWW browsers emerged, HTML language was extended to allow the pages to be designed in a more attractive manner. Nevertheless, most of the WWW content remained static and passive.

2.1.2 Web Applications

Dynamic WWW content was a great improvement on static documents. The HTML pages were generated dynamically upon user's request. The application could read data from any external source including filesystem and database engines. The specification of Common Gateway Interface (CGI) allowed applications to be independent of the underlying web server. But the CGI-based applications had to spawn a new process for every user's request and that rendered the whole application ineffective in the large deployments.

Several systems were addressing this issue, but most of them were non-portable and web-server specific. The first widespread industry standard in the area of web application interfaces was the Java Servlet API published by Sun Microsystems. The servlet technology allowed web applications to be independent of the web server. Most of the servlet engines were implemented as multithreaded servers separate from a web server. The separation of the application engine from the web server was the beginning of modern application servers.

The application server is the most important component of a server tier of the multi-tier web-application architecture (Figure 1). It provides the environment in which application components are running. Typical application server provides appropriate structures and interfaces for an application to interact with external systems and users. These interfaces include a relational database access interface, naming and directory services support, enterprise messaging interfaces, etc. The application server also provides an environment for building a graphical user interface using HTML, WML or some other presentation-oriented language.

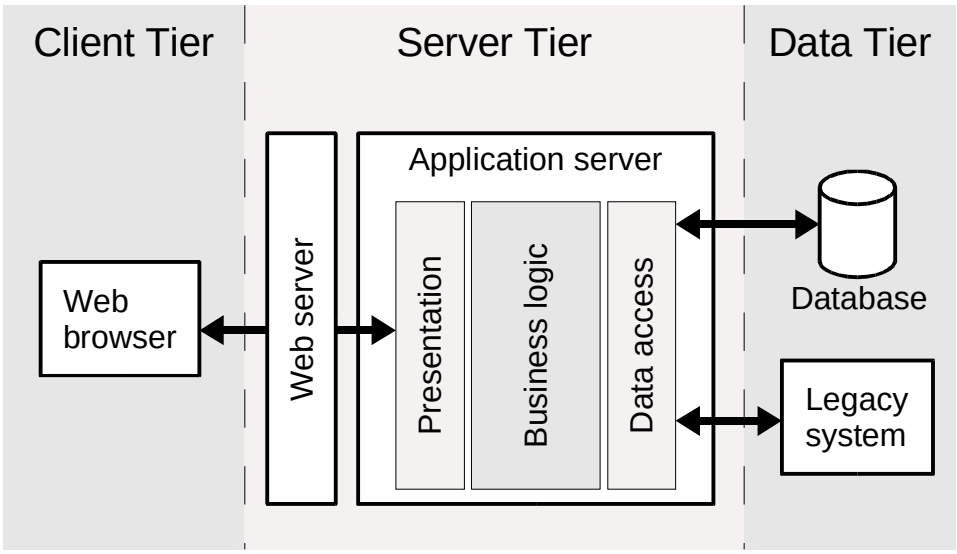

Figure 1 Three-Tier Web Application Architecture

2.1.3 Web Applications security issues

The original TCP/IP design was not concerned with the security issues [1] and the designers of web protocols followed the same model. The HTTP protocol implements only the minimal set of security mechanisms. Simple password authentication is supported by the HTTP protocol (HTTP Basic Authentication), but it is seldom used today. The most common way to secure a web application is to protect the HTTP protocol by layering it on top of the SSL-secured channel. This method is commonly referred to as the HTTPS [2] protocol.

User's credentials are supplied to the application by using an HTML form. These credentials are processed by the application login and may support almost any authentication scheme, but the most common way is to use simple static passwords. The HTML form conveying the credentials may be protected by the SSL protocol, but is frequently left unprotected. It is obvious that this weak authentication mechanism is not appropriate for most web applications.

The web application access control model is not unified and it is generaly designed in an *ad hoc* manner. Several access control models suitable for web environment were proposed in the literature [3], but none of them was practically deployed in the Internet enviroment due to the lack of web application infrastructure components.

2.1.4 Web Services

As the paradigms of electronic commerce (e-Commerce) and electronic business (e-Business) emerged, it soon became clear that a higher degree of automation of the business processes is needed in the business-to-business (B2B) transactions. Web applications cannot satisfy this requirement as there is still a need for person to operate the applications.

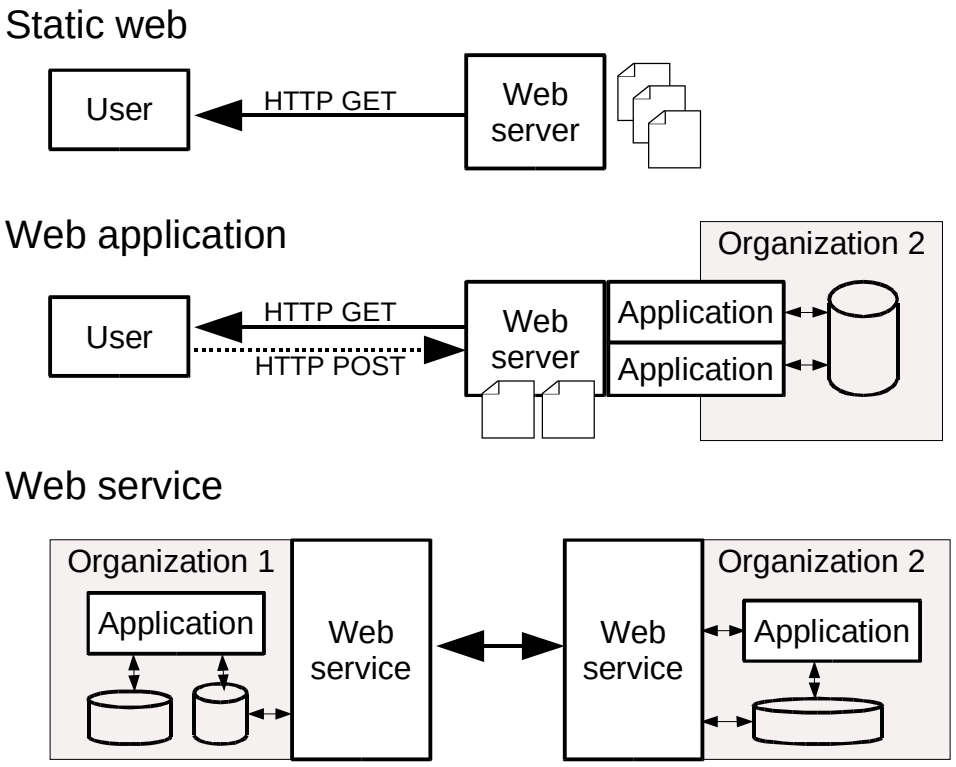

Figure 2 Static web, web application and web service

The web service technology is addressing this issue. Web services are still based on common Internet transport protocols (HTTP, SMTP, etc.), but an XML-based [4] data format is used for data representation. These data are not directly intended for end user, rather they are used by automated information processing systems. As the web service communication is based on the operating system and programming platform independent XML standard, different computing platform can communicate transparently. The Simple Object Access Protocol (SOAP) [5] is used as a primary protocol for accessing web services. The web service schematics and location are defined using the Web Service Description Language (WSDL) [6]. The static web pages, web applications and web services are compared in Figure 2.

2.1.5 Web services security issues

The early design of World Wide Web included only the simplest security mechanisms and was not designed to address the complex security issues of today's Internet [7]. There were several attempts to improve the WWW security, but until recently only the SSL protocol [8] became widely used. Most of the current web applications use the simple password authentication mechanism, managed by each application separately and protected by the SSL during the network transfer. Such protection level is sufficient for the stand-alone web applications, but the enterprise-class applications and web services needs a more sophisticated approach.

The web service non-interactive usage pattern makes the use of the static passwords inefficient even in the small and security insensitive systems. It is clear that distributed security infrastructure will be needed to secure wide range of web services.

2.2 Directory services

Traditional computing environment was composed of several stand-alone systems, each of them maintaining its own user database, authorization records and other security policy-relevant data. The growth of distributed systems required a shared repository accessible to all nodes of the system. But these repositories were still local, useable only by single system. As the requirements on the information systems integration grew, a need for a global, universally accessible directory service emerged. In 1993 the ITU-T proposed the recommendation X.500 [9], which specified a generic directory service concept. But it was soon apparent, that the full X.500 directory implementation is too heavyweight and complicated for general use. The Lightweight Directory Access Protocol (LDAP) [10] was proposed to overcome the complexity problems of X.500. The LDAP concept builds on the X.500 directory model, but specifies simpler communication protocols for directory access. The LDAP-based directories became the *de facto* standard for platform-independent general-purpose directory services.

A directory service is a vital part of the modern distributed information system. User identities, network objects, software module configuration parameters and other global data are typically stored in a directory server databases. The organization's directory tree can therefore be naturally used as a base for user identity management and digital identity infrastructure.

2.3 Internet security considerations

The security models used for the Internet environment commonly assume that the attacker can control the information flow between network nodes, while the network nodes itself remain secure. These assumptions may or may not be appropriate for some specific situation, however we will use this model in this work. We will also assume, that the intruder's goal is to attack one or more of the security properties of the protected asset. The asset's security properties includes:

- Confidentiality – protection of the information from an unauthorized disclosure.
- Integrity – protection of the information from an unauthorized modification.
- Availability – ability to deliver the information when needed.
- Authenticity – assurance that the data origin is know and authentic.
- Accountability – traceability of the subject's actions.

The Internet security mechanisms are frequently concerned only by the transport of the assets between network nodes. Although the network node itself is considered secure, we will attempt to concentrate on the mechanisms that can provide end-to-end security, if needed.

The previous work on the Internet security introduced several practical security protocols many of which are in common use on the Internet. Some of the most important of these security protocols are depicted in Figure 3 and their positioning in the TCP/IP protocol stack is illustrated. Most of these protocols are hybrid and cannot be precisely positioned in the protocol stack. Protocols of lower levels tends to be transparent to the user and provide coarse-grained security services and the protocols of higher layers provide more control on the data protection but also require more concern on the side of application users and developers.

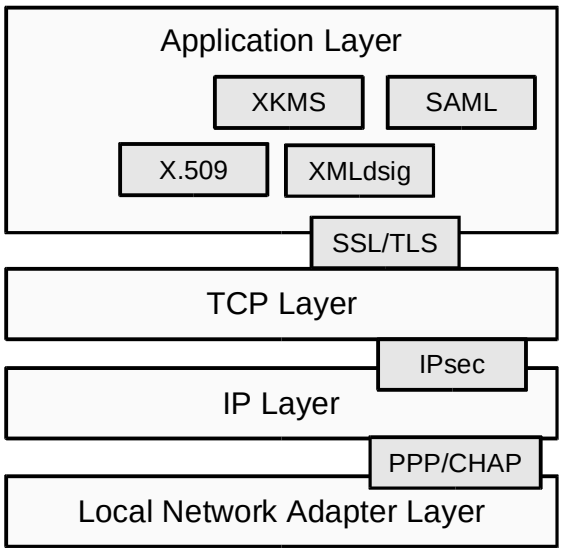

Figure 3 Internet security protocols positioning

This work concentrates on the protocols of the higher layers and special attention is paid to the protocols of the application layer. Especially protocols based on the emerging XML standard are considered, as these promise the best support for the web application and web services security.

As the Internet grows and applications take advantage of the network effect, the application complexity grows considerably. The traditional standalone web applications are being replaced by distributed application systems based on the heavy use of the web services. Appropriate security mechanisms are needed for this type of applications. Traditional password authentication and most of other interactive authentication systems are not appropriate for deployment with the next generation web services. Some methods expected to provide appropriate security services for the distributed Internet applications are considered in this work.

3 Authentication and Access Control

Information processed by today's information systems is frequently of private character and the violation of its security properties could cause a severe damage. It is clear, that the security mechanisms form a vital part of today's information processing systems and communication network architectures. Authentication and access control mechanisms are the most frequently used security mechanisms and make an important part of the overall security architecture.

3.1 Traditional Authentication Methods

Password authentication is a very ancient method of authentication. It was used in computer systems even before computer networks evolved and is still very popular. Password authentication has many security drawbacks, but its simplicity is the primary reason for its wide usage. Passwords sent in cleartext over the network are subject to eavesdropping and replays, passwords can be stolen either from the end user or from a server database and most of them can be easily guessed [11]. Eavesdropping and replay attacks on the communication layer can be prevented by employing a good encryption mechanism, but passwords can still be compromised in client or server operating systems, before they can be encrypted. Some operating systems provide password-caching features that can be misused by an attacker to read a password cache and get all stored passwords.

It is not possible to use an encrypted channel to protect passwords in every circumstances and the static character of passwords causes different kinds of problems nevertheless. To overcome these issues, several One Time Password (OTP) schemes were proposed. The most popular one time password scheme was the S/Key scheme, developed at Belcore [12] and later standardized by IETF [13]. This scheme is based on the repeated application of the one-way function to the secret value to get the one time password sequence. Passwords from this sequence are used in a reverse order for authentication, each used only once. Password sequence contains fixed amount of one-time passwords and must be restarted when all of them have been used.

Several token-based commercial one-time password schemes appeared on market in last years. One of the most wide-spread systems is the RSA Security SecurID. Implemented as the hardware card, key fob or the software application, SecurID generates 6 to 8 digit numbers in regular time intervals (30 or 60 seconds). These numbers can be used as one-time passwords for the authentication. The algorithm generating these sequences has not been published, but there were some successful attempts on its reverse engineering [14]. The shared secret value for the SecurID is only 64bit long, which seems to be too short for today's security requirements.

The challenge-response authentication scheme [15] has similar properties to one-time password schemes. Strictly speaking, one-time password schemes are only special instances of the challenge response schemes with fixed challenge information (sequence number, time instant, etc). The challenge-response scheme client takes some information (challenge) from the authentication server, processes it with the user-supplied secret value (password, shared secret) and returns the processed information (response) to the server. By considering the sent challenge value and received response the server can determine if the user knows the correct

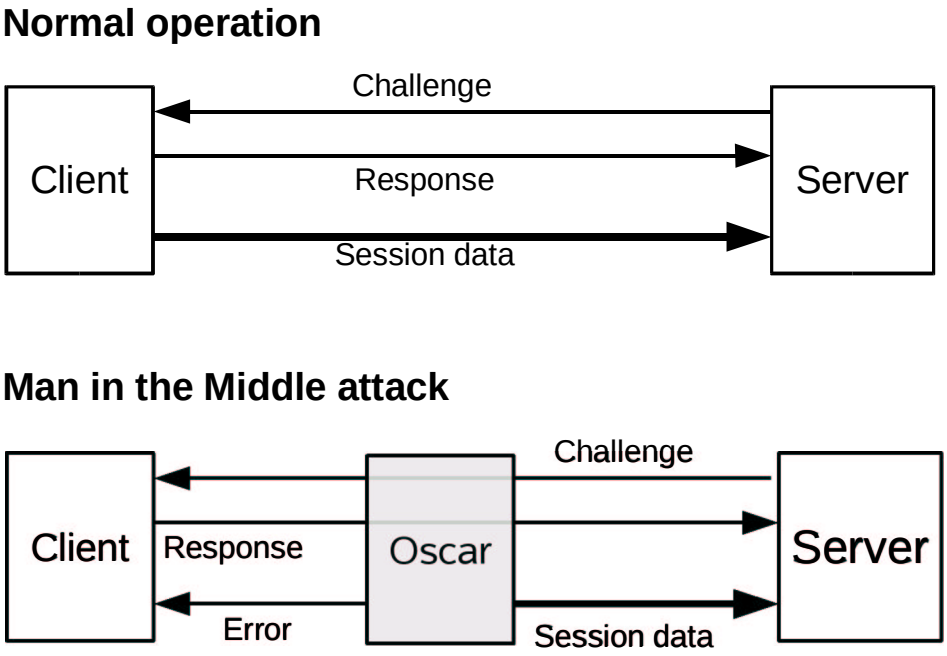

Figure 4: Man in the middle attack on challenge-response scheme

secret value. The most widely used challenge-response method is CHAP [16], used as a part of PPP protocol.

Both one time password schemes and challenge-response schemes in general have some common security drawbacks. When used in a plain TCP/IP environment, connection data can be manipulated after a successful authentication takes place. In this case the attacker does not need to attack the authentication scheme directly. If the authentication scheme accepts the secret value (seed) as a plain password that is chosen by the user, such a scheme is vulnerable to the dictionary attacks. Most of these schemes which are in practice are vulnerable to the man-in-the-middle attacks (Figure 4). These attacks can be prevented only by pre-authenticating the server to the client by using other independent methods and explicitly authenticating the transported data.

The common way of securing any password scheme is the use of an encrypted channel with an authenticated server. The Transport Layer Security (TLS) and Secure Shell (SSH) are the two most widely used methods to achieve this goal. Both of these protocols provide an encrypted channel to secure the password authentication from eavesdropping. Server authentication is achieved by employing asymmetric cryptography, either by ad-hoc methods in SSH or by using a X.509-based PKI in TLS.

3.2 Public Key Based Methods

Both the static passwords and the symmetric cryptography-based authentication methods often exhibit poor characteristics when deployed in the large networks. Poor scalability and the requirement of the secrecy of shared information limits the efficiency of these systems. On the other hand, asymmetric cryptography

techniques can be utilized even in the global environments and when used appropriately they do not suffer from such problems.

3.2.1 Public Key Infrastructure

The Public Key Infrastructure (PKI) is a set of methods and formats for the management of public keys and all related data. PKI uses asymmetric cryptography methods to achieve its goals. Each entity in PKI has at least one asymmetric key pair. The identity of the entity is bound to the entity's public key by the Public Key Certificate (PKC). This certificate is a data structure that contains the entity's identification, certificate validity period, the identification of the certificate issuer and any other data describing the certificate and its use. The PKC also contains the entity's public key and the entire PKC is signed by the certificate issuer's private key.

There are some infrastructures that do not impose any limits on which entity may issue certificates and which may not. These infrastructures are called 'Web of Trust' and their trust structure forms a generic directed graph. A good example of this kind of structure is PGP. Other infrastructures limit certificate issue privileges only to selected entities. These entities are called Certificate Authorities (CA) and their role is the management of certificates issued to End Entities (EE). Certificate Authorities may issue certificates to each other, expressing trust relationships. Trust structure of this PKI is hybrid. The trust relationship between the End Entities and their Certificate Authorities forms a directed tree, but the relationships of different CAs can form any generic structure.

3.2.2 X.509 Certificates

The international standard for the Public Key Certificate format is based on ITU-T X.509 recommendation [17] and is widely used in both enterprise and Internet environments. Other formats evolved as internal parts of specific applications, but use of these proprietary formats yields in favor of X.509 certificates. As X.509 is both official and *de facto* industry standard, the rest of this document will cover X.509-based PKIs only.

The certificate authority approves certificate validity by its signature. The situation occurs that the certificate’s validity must be terminated, for example due to private key being compromised. Each X.509 certificate contains a fixed validity period, but in the practice it is not possible to wait untill the end of the validity period to invalidate the certificate. Some mechanism must exist to revoke the certificate anytime during its validity period. The certificate authority publishes a list of certificates for this purpose, that had to be revoked before the end of their validity period. This list is called the Certificate Revocation List (CRL) and is published at regular or irregular intervals. CRL is protected by the certificate authority signature so it can be stored in an insecure environment. The entity that is checking the certificate validity should locate the appropriate CRL and check if the certificate is not listed there. The CRL method of the validity checking may be insufficient for certain applications that require on-line certificate validation. These applications could use on-line validation protocols such as Online Certificate Status Protocol (OCSP) [18].

Version 3 of the X.509 recommendation allows the use of certificate and CLS extensions. These extension can be used to specify an additional information, such as the certificate use constraints, subject and issuer alternative names or almost any other information. Use of the extensions greatly enhances flexibility, but different, non-interoperable, implementations of the same basic mechanisms appeared. Each of these implementations understands a different set of extensions and even if they agree on a common set, they interpret the extension values in a different way. To promote interoperability of certificate processing systems, the national organizations and standard bodies publish X.509 certificate profile documents. These certificate profiles specify the exact meaning of certificate extensions, rules for certificate processing and so on. The IETF published the X.509 profile for use in the Internet environment [19].

3.2.3 Transport Layer Security Protocol

Public key certificates can be used in many communication systems on the Internet. The most common communication protocol in use today that employs X.509 public key certificates is the Transport Layer Security (TLS) Protocol [20]. It was originally developed at the Netscape Communications corp. in 1994 as the Secure Socket Layer (SSL) protocol. The first publicly available version of SSL was version 2 [21], which suffered from several major problems [22]. That version was later updated to version 3 [8] with most of the SSv2 security problems eliminated. The SSL protocol version 3 was taken by IETF as the base for the TLS protocol version 1.0. The TLS and SSLv3 specifications have some minor differences that implementers should take care of to assure compatibility. Compatibility is not straightforward, but protocol versions could be correctly detected by examining initial protocol messages. The SSLv2 is not compatible with SSLv3 nor TLS, but implementation could support all these three protocol on the same TCP port. Nevertheless implementers are encouraged to use the TLS protocol or at least the SSLv3 protocol instead of SSLv2 whenever possible.

The TLS protocol consists of two internal layers and several subprotocols. Overall TLS structure is depicted in Figure 5. The lower TLS Record Protocol is used for transfering the protocol data units across the communication channels. The higher layer is dedicated for the session parameters establishement, management and

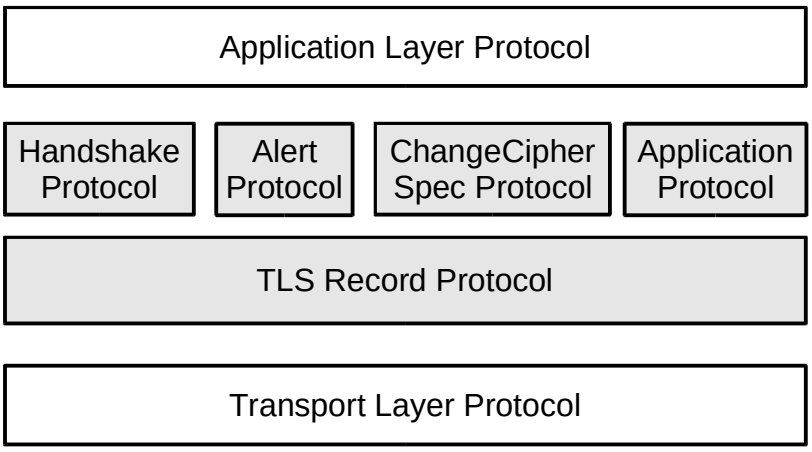

Figure 5 TLS Protocol Structure

alerting. The application subprotocol is used for transfering user data, provided by application layer protocols.

The TLS protocol does not depend on any specific cryptographic algorithm. The communicating parties can negotiate the best common cryptographic algorithm suite for secure communication. The protocol is asymmetric, the client is the connecting party while the server passively listens for connections. TLS supports several authentication modes:

- **Total anonymity**. Neither the server nor the client are authenticated, no key material origin is assured. This authentication mode is possible with TLS but its use is strongly discouraged.
- **Authenticated server.** The server authenticates to the client by presenting its public key certificate and providing proof of possession of the appropriate private key. Client is still anonymous, but the key exchange can be accomplished securely.
- **Mutual authentication.** Both server and client are authenticated to each other by using their respective public key certificates and appropriate proofs of possession of the private keys.

The most frequently used TLS mode today is the authenticated server mode. The server has an X.509 certificate for its fully-qualified domain name (FQDN) and the client (web browser) has a list of trusted certificate authorities. The client authenticates the server using its X.509 public key certificate and both the server and the client negotiate a session key. When the secure communication channel is set up, the client authenticates using a password, challenge-response system or whatever mechanism is appropriate. This client authentication must be processed on the application layer, the TLS layer is not aware of it happening.

Password authentication cannot be considered “strong” even if it is protected by a partially authenticated encrypted channel. However, the TLS protocol supports mutual authenticated mode with both client and server mutually authenticated using strong authentication. Both the client and the server must posses X.509 certificates for their public keys and the corresponding private keys. These certificates are exchanged during the initial protocol handshake and the appropriate proofs of possession of private keys are presented by both parties. When this authentication mode is used, there is no need for the client to authenticate at the application layer. The application layer only makes authorization decisions.

The total anonymity mode is included for backward compatibility only and for any obscure application that may require it. The key material origin is not authenticated in this mode and the Man-in-the-Middle attack is possible. Use of this mode for whatever reason is not recommended.

If TLS is used in any authentication mode, it provides only short-term security for the transported data - TLS protects and authenticates data on the communication channel only. If the data is stored on the target system, they are no longer protected. Even if the data block was received by the server in a mutually authenticated TLS connection, server is not able to provide any proof of data origin to the third party. Use of standalone digital signatures is recommended in addition to TLS to achieve such long-term protection requirements.

The most common practical use of the TLS protocol is to secure a WWW communication. It is called HTTPS [2] and it is essentially HTTP protocol communicating over a TLS-secured channel. Most of the connection-oriented protocols can be modified in a similar way to use the TLS layer for protection. Use of TLS to secure IMAP, LDAP and other TCP-based protocols is becoming quite common on the Internet.

3.2.4 Digital signatures

Session-based authentication provides good security properties for the interactive tasks, it is ideal for UNIX shell access or a WWW application. However, sometimes there is a need to authenticate the data origin and this authentication status needs to be presented to the third party. If such a need exists, the presented document has to be accompanied by some kind of authentication information, which will serve as the proof of the data origin. Digital signatures

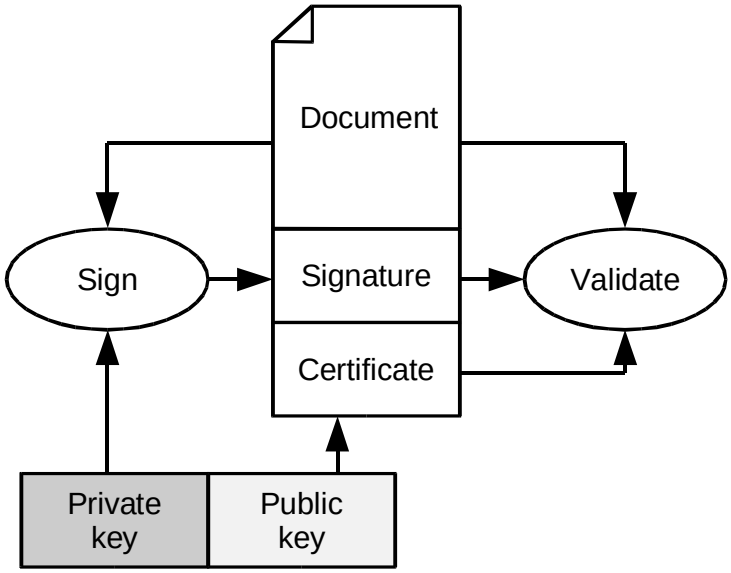

Figure 6: Simplified digital signature system

provide a suitable mechanism for such data origin authentication. Digital signatures are based on the public-key cryptography. The signature is a function of the document content and the signer's private key and can be verified by the signer's public key. As signature is a function of a document content, it assures document integrity. If a signer has a public key certificate it can be used to prove the signer's identity and the document origin. The simplified digital signature system is illustrated in Figure 6.

Digital signature methods can provide long term authenticity for signed data. Most of the cryptographic protocols based on public-key cryptography employ digital signatures internally for the data authenticity proofs. One of the first practical uses of digital signatures for the long-term data authenticity were protocols for securing electronic mail – PGP [23], PEM and S/MIME [24]. PGP and S/MIME are still the most widely used electronic mail security protocols used today.

3.2.5 Digital signatures and the real world

The digital signatures have similar properties to the handwritten signatures and the trends are for the digital signatures to be used as their equivalent. However, if digital signatures have to fulfill all the security requirements, simple digital signature schemes are not sufficient. One of the most serious problems with digital signatures is non-repudiation – the party that signed the document must not be able to later deny signing that document. When a simple digital signature scheme is used, a dishonest party could sign a document and send it to the recipient. Then the sender waits for the recipient's action and when the recipient awaits a payment, he claim that the private key has been compromised. The dishonest party could state that the signed document was not signed by himself, but by someone else who had stolen his private key.

It is apparent that timestamps are needed in this situation. If the recipient of the message could attach a trusted timestamp to the document, he can prove that the document was signed before the claimed key compromise and that the signature is therefore valid. For the timestamp to be secure and trusted, it must be issued by a trusted authority – timestamping authority (TSA). IETF proposed a Time-Stamp Protocol (TSP) [25] for use in the interaction with TSA.

Public key certificates are used for the key-holder's identity validation in both digital signature and encryption scenarios. An easy solution would be to use the same key pair for encryption and digital signatures. However, there are limitations to this setup. Some governments or organization policies may limit the use of encryption or there may be a need to use stronger (and therefore slower) authentication than encryption. For these and other reasons it may be reasonable to include two public keys in a public key certificate, one for encryption and the other for signatures. The application can then choose the appropriate key pair to use.

3.2.6 XML Key Management Specification

The X.509 PKI is a general-purpose, flexible and comprehensive infrastructure, but it could soon become very complex. The infrastructure complexity impacts especially the certificate authority systems and clients. Certificate location and verification is not an easy task even for a full-featured thick clients, not to mention portable devices and appliances. The situation gets even more complicated as more than one PKI have to be used. The clients would need to understand all the details of every PKI system used.

The XML Key Management Specification (XKMS) [26] is an attempt to solve the PKI complexity issues. Simple XML-based protocols are defined for interaction with key management services. These services should perform all the necessary complex PKI interactions on behalf of the client and return only the final results (Figure 7).

The Key Information Service Specification (X-KISS) defines a XML-based communication mechanism used for the interactions with the *trust service*. The Trust Service has two tasks: location of the necessary key material (*locate* service) and validation of a key material (*validate* service).

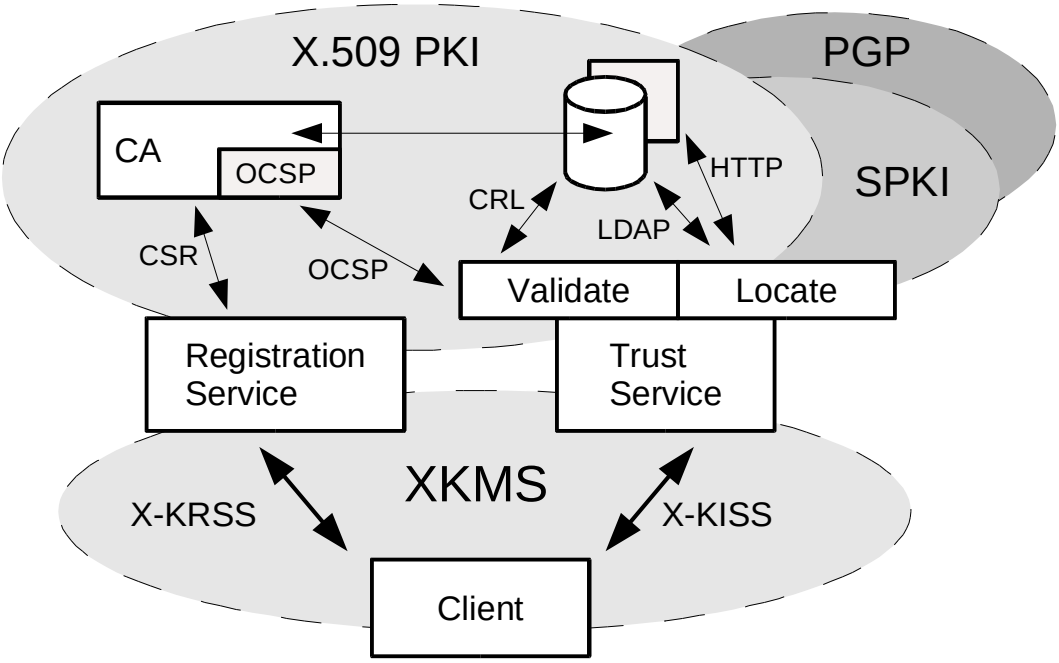

Figure 7 The XKMS interaction with other PKI systems

The Key Registration Services Specification (X-KRSS) describes a protocol for registration of a key material. The X-KRSS is used for the interaction with the *registration service*, that acts as an agent for key registration, certification and key–recovery services in different PKI systems.

The primary objective of XKMS is to off-load key management functions necessary for processing XML digital signatures [27] and XML encryption [28] from the client.

3.2.7 Privilege Management Infrastructure

Public key certificates bind the identity of a person to their public key. The identity of a person is not just the name of the person; it may include his role in an organization, date of birth and so on. Some of this identity information may be considered private, and people may not be willing to present them freely in their public key certificates. However, some authorization decisions are based on this non-public identity information. For example access to a corporate intranet has to be granted only to the corporate employees, access to some sites is limited to users over 18 and so on.

Each person may have multiple certificates for his public key. One may be his personal (citizen) certificate, another his employee certificate and yet another his community certificate. The person then selects the appropriate certificate to access different services. But this setup requires different certificate authorities to operate: citizen CA, corporate CA and community CA in this example. Each of these certificate authorities has to verify the identity of the person.

Another approach uses one personal public key certificate and adds the other certificates that approve the person's attributes. These certificates are called Attribute Certificates (AC) and are used to bind the person's attributes to his identity.

The person's attributes can specify the person's privileges, status, organizational role or any other information that is temporarily associated with that specific person. The attribute certificates are issued by Attribute Authorities (AA), which may or may not be the same as the certificate authorities. The attribute authority does not need to physically verify the person's identity. The person's identity is proved (if needed) to the attribute authority by presenting a public key certificate. The attribute authority needs to verify that the person has the specified attribute and that fact is certified by issuing an attribute certificate.

The attribute certificates can be used to make the authorization decisions at the authorization enforcement points. A person accessing a protected resource has to provide their public key certificate, appropriate attribute certificates and proof of possession of the public key. The authorization enforcement point verifies the person's identity using a public key certificate and then verifies their authorization by examining and verifying attribute certificates. The person needs to provide only a minimal set of attribute certificates to gain access to the resource. The person's privacy is maintained as much as possible.

The attribute certificates generally have much shorter lifetimes than the public key certificates and therefore Privilege Mangement Infrastructure (PMI) can be more flexible than plain PKI. Attribute certificates may be used for short-term authorization for access to services or they may be used for longer-term role assignment.

The attribute certificate framework and its use with X.509 PKI is defined in ITU-T X.509 recommendation [17], but no specific attributes for general use are defined there. Implementation details of specific PMI depend on the implementer’s choice. It can be expected that the attribute certificate profile specifications [29] emerge as the PMI concept gains wider acceptance.

4 Digital Identity

The Internet connects myriads of different hosts, organizational networks, enterprises, service providers, etc. Each of these systems is trying to enforce its own security policy. A typical Internet user's interaction is not limited to one organizational network; he makes use of several different systems that cross organization boundaries. That means, several security policies are enforced during the user's Internet session. A typical outcome of this is the annoying fact that the user must authenticate separately at each site that he visits to obtain their full privileges. At the same time, the Internet is becoming more complex and the organizational networks become more open to the external user community. Each of these factors contribute to greater inconvenience for a typical user.

The user must maintain several identities throughout the Internet – web site registrations, electronic banking access, credit card information, accounts on B2C e−commerce sites, etc. Access to the Internet sites is typically protected by a simple password. To avoid the misuse of access credentials by an unscrupulous site it is necessary to choose a different password for each site. A password list soon becomes long and inconvenient, needs to be stored somewhere, or users just take the risk and choose the same password for each site. But both of these approaches are far from being ideal.

4.1 Single Sign-On

Multiple authentication has been a problem in heterogeneous enterprise information systems even before the Internet gained wide acceptance. Solutions to this problem range from simple password lists encrypted with a master-password to the use of Kerberos [30] and similar security systems. Many proprietary solutions were developed by commercial companies and most of these products were endorsed as Single Sign-On (SSO) systems. They allowed the user to login once and then use all of his resources of all enterprise systems during his session without re–authentication.

Single Sing-On applications developed for closed enterprise environment are not suitable for the Internet environment. It is quite clear that no proprietary solution will work in large-scale Internet deployment. Kerberos and its modifications were designed for intra-organization use and are not suitable for global Internet eighter, because of the high key management overhead. The use of new XML-based technologies looks very promising in this field.

There are several proposed single sign-on architectures for the Internet, but most of them follows a similar architectural approach [31] [32] [33] [34]. The user's identity and authentication information is maintained on an *identity manager* server (Figure 8). The user first authenticates with the identity manager and then accesses the resource serviced by a *resource manager*. The resource manger may be a part of a different organization or domain than the identity manger, but there must be a trust relationship between the two. The identity manager provides authentication assertion to the resource manager on request. This assertion states that the user completed the authentication procedure with the identity manager and that it now knows the identity of the user. Single sign-on schemes do not dictate the initial authentication of the user; the specific authentication scheme used is out of scope of the single sign−on system and depends on local requirements.

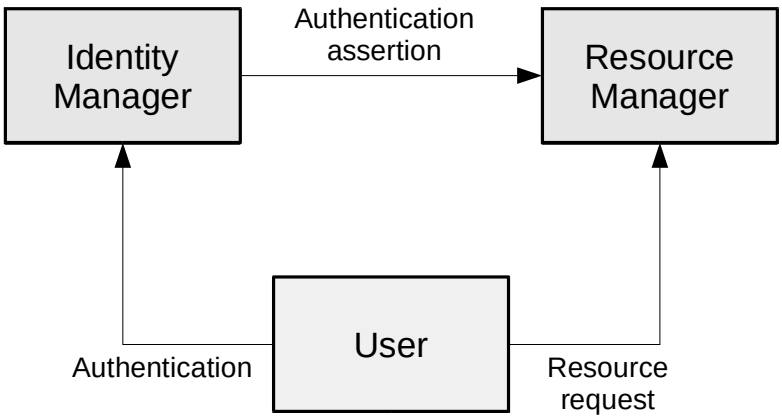

Figure 8: Single sign-on system architecture

Most of the single sign-on schemes are concerned with a web user – a user equipped with a web browser accessing a web page or application. Information transfer between the identity manager and the resource manager is achieved by various mechanisms based on the HTTP protocol.

The one potential problem with SSO systems is a liveness problem. A user starts a session by authenticating with a identity manager and than may access resources on resource manager's sites. When the user requests resource at some later time, a doubt may arise whether it is still the authenticated user that is requesting a resource. Resource manager could request reauthentication of user, especialy if some valuable resource is being requested. Identity manager should take this consideration into account and probably support several authentication schemes of different security assurance level. Then the trust given to a specific authentication assertion would be a function of the authentication security level, the time of last successful authentication and the value of the resource requested.

4.2 User profiles

The Internet identity mechanisms provide much more than just a solution to the multiple authentication problem. One of the primary concerns of the digital identity is the management of the user's personal information. The user's real name, address, credit card number, employer, e-mail address, telephone number, etc. form a user profile. User profile is a collection of information that the user wants to share with the selected destination sites. The secure sharing of user profile information is a task of the identity manager. The identity manager responds to the user profile requests from the destination sites. This approach will provide the current information to the destination site without the inconvenience of filling out registration forms, etc. However, the user must be able to specify which destination sites are trusted for which parts of his profile. Identity managers should provide appropriate user interfaces for this purpose.

4.3 Security Assertion Markup Language

Security Assertion Markup Language (SAML) is a specification of the syntax and semantics of the security assertions encoded in XML [4]. The SAML specification also defines the format of requests and responses, SAML binding for other protocols and appropriate XML schema. The SAML specification [35] was published by the Organization for the Advancement of Structured Information Standards (OASIS) and is currently in the standardization process.

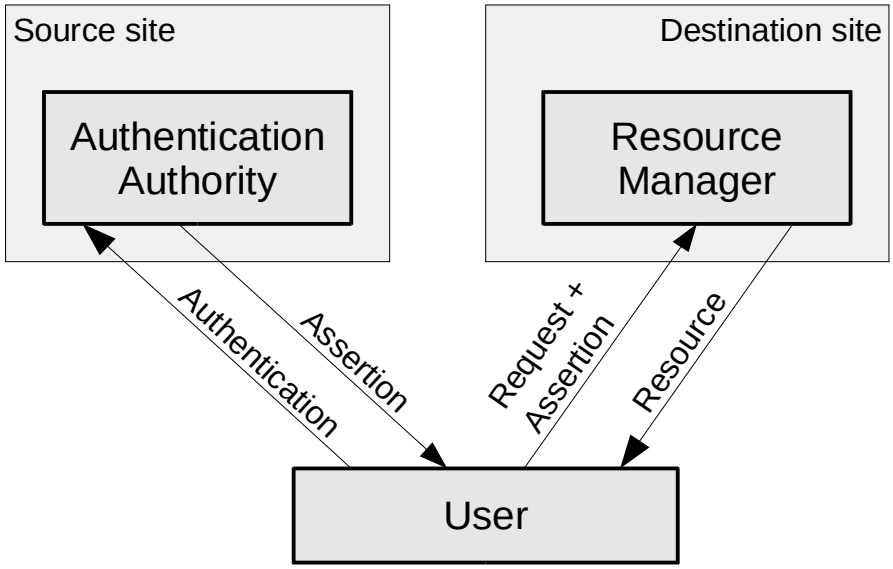

Figure 9: Simple SAML scenario

SAML assertions can be used to assert specific characteristics of an entity. They can assert the authentication status, authorization status or binding of an attribute to an entity. Assertions are issued by the appropriate authorities and used by security policy enforcement systems. Authentication assertion issued by the system that the user logs into can be used by another system to grant the user a resource without any re-authentication. This simple scenario is illustrated in Figure 9. The user authenticates using the source site authentication system. The authentication authority then issues an authentication assertion to the user. The user then requests a resource on the destination site and includes the issued assertion with the request. The destination site examines the provided assertion and makes a decision based on the destination site's security policy. If the destination site trusts the source site to authenticate the user properly and the user accessing the resource is authorized to do so, the destination site grants the user a resource without any re-authentication.

The resource manager on the destination site must implement a full policy decision mechanism. This may be inappropriate in larger and more complex systems with several resource managers and policy enforcement points. The destination site must know the name of the user accessing the resource, which may also be undesirable. The membership of the specific user group or role may be sufficient authorization for accessing the resource. These shortcomings are addressed in a more complex approach depicted in Figure 10.

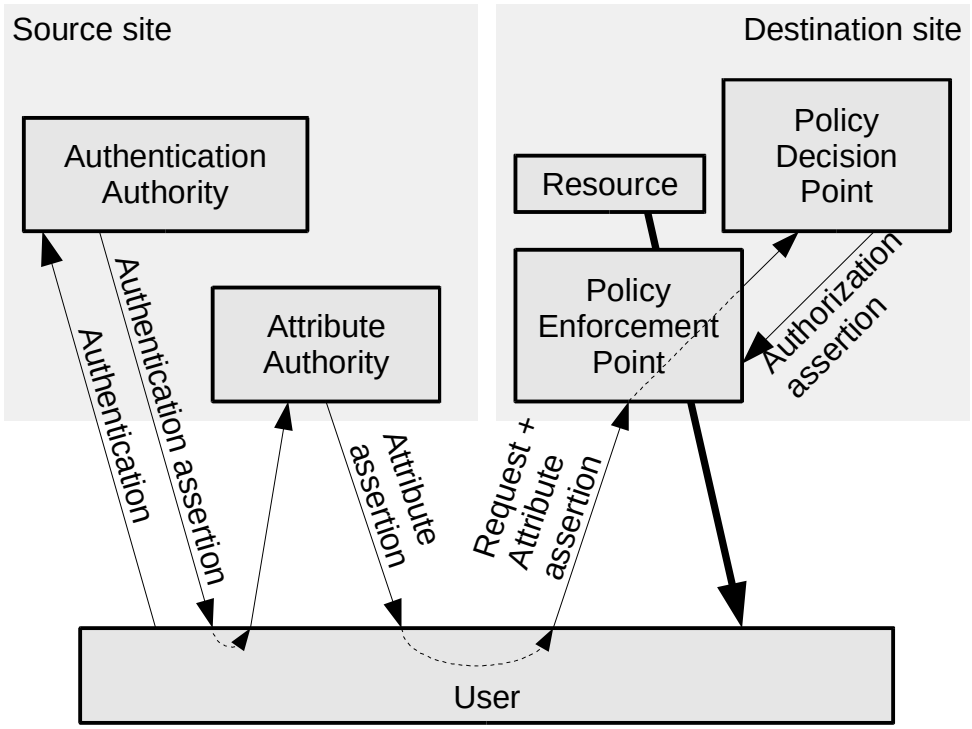

Figure 10: Complex SAML scenario

User is authenticated by a source site authentication system and the authentication authority then issues the *authentication assertion* to the user. The user sends this assertion to the attribute authority and requests the appropriate *attribute assertion* for their organizational role. Then the user can make a request to the destination site's protected resource. This request is accompanied by the attribute assertion. The destination site receives the request, but it must first be processed at a policy enforcement point to check its authorization. The policy enforcement point checks that the requests conform to the security policy by a request to the policy decision point. The policy decision point consults the security policy database and returns the *authorization assertion* for the resource request. The policy enforcement point then proceeds with satisfying user's request.

Most of the current information systems in development follow a web application pattern for the user interface. The user is equipped with a web browser that uses HTTP protocol to access the user interface. Web services are also commonly used today as a platform independent, interoperable way of inter-process communication. To accommodate this situation SAML specifies bindings and profiles for common usage patterns. SAML binding and profile specification [36] defines SAML SOAP binding and two web browser SSO profiles. The SSO profiles define a scenario of a single sign-on mechanism in a web environment. Two ways of passing SAML assertions from the source site to the destination sites are defined:

- **Browser/artifact profile.** The user's browser is redirected from the source site to the destination site and a unique 8-byte identification of SAML assertion is provided in the query string of the redirected request. This ID is called *SAML artifact* and is used by the destination site to dereference the original SAML assertion directly from the source site.

- **Browser/POST profile.** The SAML assertion is transferred from the source site to the destination site as the POST from a HTML form.

Specification of SAML assertion is based on XML, which makes it platform independent and can be easily processed on almost any application platform today. Together with other XML standards like XML Signatures [27] it may be extended to provide a flexible and reliable single sign-on mechanism. However, SAML alone is not an identity system, it just provides a standard and interoperable way of security assertion exchange. It may be used as part of larger systems, as can be seen in [33] and [34].

4.4 Digital identity systems

Several different systems for Digital identity management have been proposed, some by commercial companies and others by academic organizations. In the next sections we will discuss the most widespread and the most promising of these identity systems.

4.4.1 Microsoft Passport

Microsoft Passport is a centralized identity system based on symmetric cryptography. The heart of the entire system is a single system located in the passport.com Internet domain. The identity information of all Passport users are stored within this single system. Every user is assigned a unique 64bit identifier called PUID. This identifier is sent to the resource manager in the form of an encrypted "ticket". The authentication sequence of the Passport system is depicted in Figure 11 and consists of these steps:

Step 1: Initial resource request. The user requests a protected resource. The resource manager looks for ticket in the user's request.

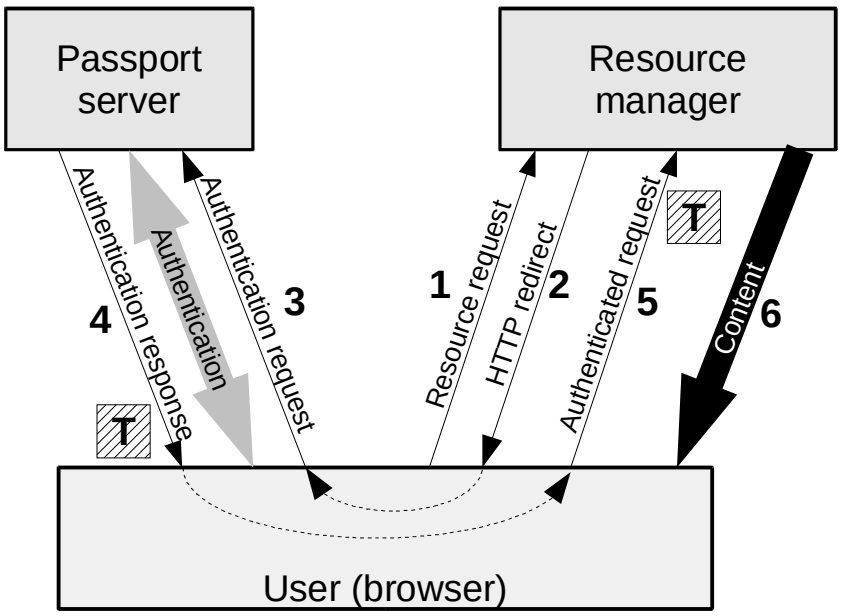

Figure 11: Microsoft Passport single sign-on

Step 2: Redirect to Passport. If the resource manager is unable to locate a valid ticket, it redirects user's request to the passport server using HTTP redirect.

Step 3: Passport authentication request. The request redirected by the resource manager is considered to be an authentication request by the Passport server. The Passport server initiates the password authentication procedure if needed.

Step 4: Authentication response. The response is sent back to the user, that contains a Passport ticket **T**, which will be sent to the resource manager for authentication.

Step 5: Authenticated resource request. The user's browser is redirected to the resource manager site with the Passport ticket included in the request.

Step 6: Content delivery. The resource manager examines the ticket in the authenticated resource request and if successful, provides the user with the desired resource.

Thus far the situation and mechanisms are clear, but Microsoft documentation [32][37] is quite incomplete in details. The exact content of the Passport ticket is unknown, as well as many other details. But it is quite clear that the ticket contains the PUID and an undocumented form of timestamp, encrypted with 3DES algorithm using a symmetric key shared by the Passport server and the resource manager.

Microsoft Passport has several security limitations and drawbacks, some of these were already pointed out in the literature [38][39]. The most critical Passport architecture problems are summarized here:

- Global centralization. The Passport server is centralized on a single system. Even if this system is made highly redundant, it could be a single point of failure. Trust to the service provider is another concern here. While using the Passport server for authentication, users and resource managers have to trust a single organization to behave correctly. The distributed architectural approach would be more suitable in this situation.
- Lack of documentation. Microsoft Passport technical documentation does not provide sufficient technical details for sufficient independent evaluation of the Passport’s single sign-on protocol. Details provided by Microsoft and found by independent researchers throws doubt on the Passport’s security.
- Passport uses a simple password authentication mechanism. Passwords are subject to easy theft and dictionary attacks and are not secure for most of the real-world applications. Passport provides “strong credentials sign-in” which is just another four-digit password with stricter usage policy. Password authentication alone cannot be considered sufficiently secure for today’s Internet applications.
- Sensitive information is protected by 3DES symmetric encryption algorithm. The use of symmetric cryptography for a global-scale system such as Internet identity system may soon become difficult. Symmetric key management tasks such as key material renewal may become infeasible in large deployment with a large number of resource manager sites. An additional mechanism based on asymmetric cryptography should be used for key management purposes.
- The passport server makes use of the SSL [8] protocol to secure some parts of the communication sequence. The SSL protocol is used in the authenticated server

mode, which relies on the PKI infrastructure for server authentication. While Passport relies on PKI indirectly through SSL, it takes no other advantage of such an infrastructure already in place. Passport should integrate with PKI more tightly to overcome some potential problems described before.

- The Passport protocol is not standard-based. Interoperability with other security systems could be a major problem.

Passport's single sign-on protocol was the first deployed Internet-scale system in 1999. Since then, the security level provided by Passport was found to be insufficient for today security needs. Passport authentication could be deployed as a short-term solution, but an implementer looking for a secure single sign-on solution should consider the use of other systems.

4.4.2 Liberty Alliance project

The Liberty Alliance Project is a group of industry and non-commercial organizations whose objective is to prepare an open standard for the network identity systems. Decentralization and openness are the main goals of the alliance, their effort aims at providing *federated identity*. The alliance was founded in September 2001 as a reaction to the Microsoft Passport project and the digital identity market needs.

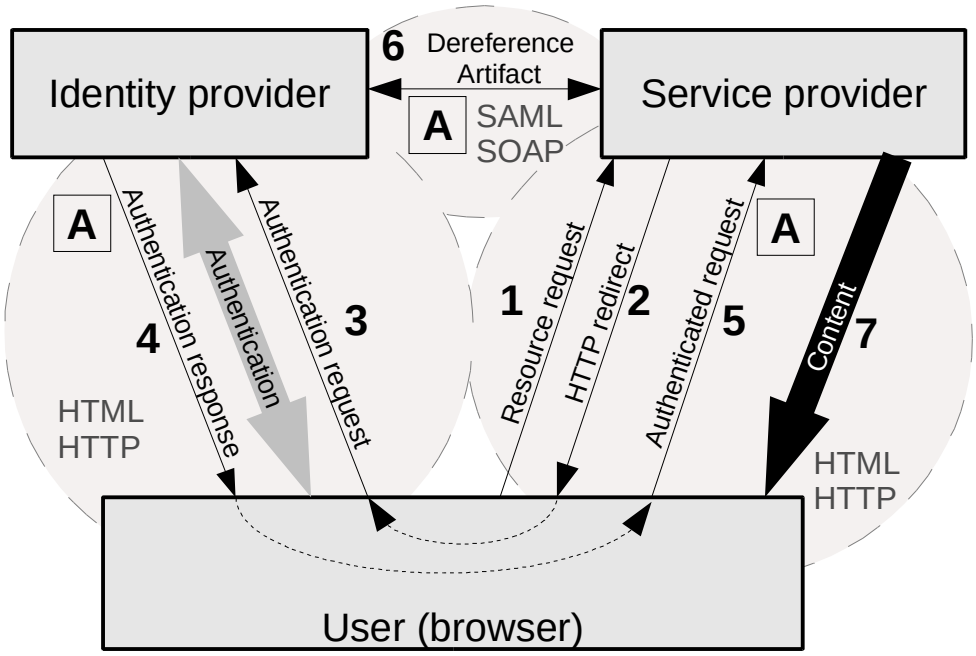

Figure 12 Liberty Browser/Artifact profile for single sign-on

The Liberty digital identity architecture [34] is heavily based on the Security Assertion Markup Language (SAML). SAML is used for expressing and transporting security assertions between Liberty-enabled sites.

The Liberty architecture is focused on the 'browser user' - a user equipped with a standard web browser software. Series of profiles is specified [40] to allow a browser user to take advantage of the single sign-on system only by using

a standard, unmodified web browser software (Figure 12). Additionally, the Liberty-enabled browser and proxy profile is specified to allow more intelligent end devices to take part in Liberty protocol directly (Figure 13). These Liberty-enabled should use special Liberty headers in HTTP communication and should be able to exchange SAML assertions over the SOAP protocol directly.

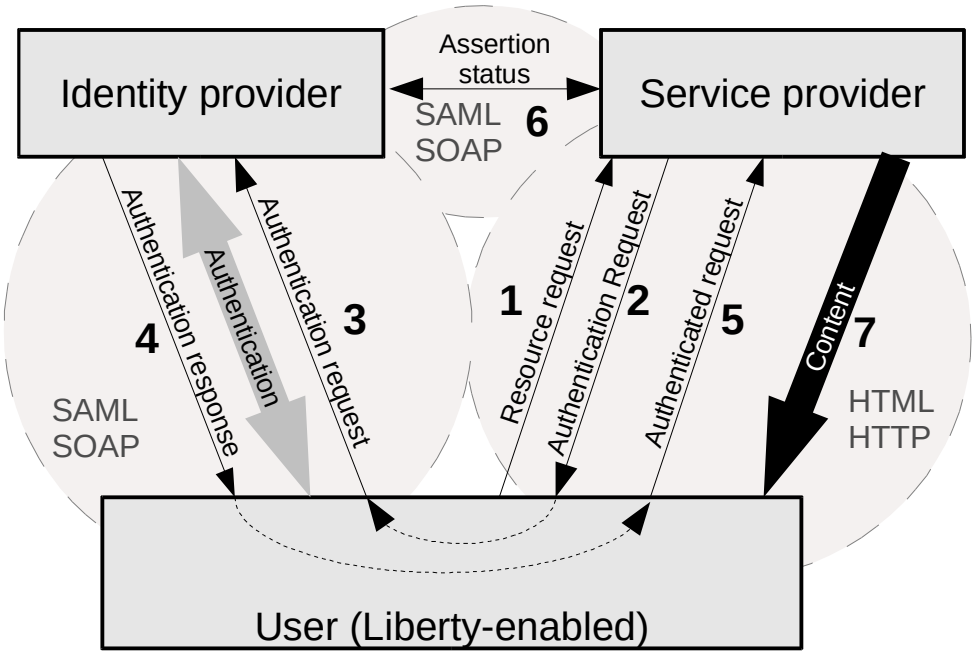

Figure 13 Liberty-enabled client and proxy profile for single sign-on

The Liberty specifications recommends SSL/TLS for a channel security and employs the SAML message signing for a message security. No global namespace for user identifiers is required by the specifications.

4.4.3 Other identity systems

Several others identity systems were proposed to manage the Internet user identities.

IDsec [31] is a virtual identity specification, which is part of the DotGNU project. It specifies the architecture for a distributed network identity system that makes use of certificates for single sign-on and user profile distribution. The IDsec mechanism is specified on the architectural level only, no specific interfaces or protocols are defined.

Shibboleth [33] is a project of Internet2/MACE, which aims at the development of an inter-institutional resource-sharing system. This project includes a framework for a single sign-on system based on SAML.

PingID and **XNS** are other projects that are developing digital identity systems. These projects are led by commercial companies and they lack sufficient public technical documentation at the time of this writing.

5 Conclusion

Traditional security mechanisms used in the Internet environment today have been found inadequate for the future needs. System implementers should choose the authentication and access control mechanisms that can integrate with a larger security infrastructure.

Public Key Infrastructure based on ITU-T X.509 recommendation has been found to be a suitable base for security mechanisms. However its heavyweight nature may not be suitable for the end-user applications. The XML Key Management Specification (XKMS) has been found as a acceptable mechanism to off-load heavyweight X.509 processing from the end-user systems and applications and thus lowering the entry barrier of the PKI processing. But even the server-side PKI processing may not provide an appropriate user experience and flexibility and X.509 could be used only as the heavyweight infrastructure for asserting server identity while users would use lightweight and more flexible security technologies. Digital identity systems could be used in that situation and could provide appropriate security services in the future.

Security Association Markup Language (SAML) was described as a base for the digital identity systems. However SAML usage is not limited to these systems and may be used as a firm base for the Internet-scale security infrastructure. SAML assertions can be employed as the security tokens in web services exchanges and thus providing a higher security level and functionality integration.

Digital identity systems, which may be used to extend user authentication to sites beyond organizational control, were described and found suitable for use in conjunction with the emerging world of web services. We believe that these identity services will be frequently used for site-to-site Internet authentication in the near future and a broader Internet security architecture will be based on digital identity in the longer term.

However, the digital identity mechanisms are too young to be considered complete at this time. As these technologies deal with the personal data of high value, the security and privacy concerns are vital. Usage of the personal information should be governed solely by its owner and the distribution of this information should be controlled as tightly as possible. But the control over information in highly distributed systems is not a trivial task. The digital identity technologies address some of the fundamental privacy issues, but many of these still remain unsolved. Especially mechanisms for secure replication, caching and data consistency are not yet addressed.

The digital identity mechanisms provide an authentication framework, but the authorization concept for distributed systems is not yet fully addressed. The SAML specification provides a definition of authorization statement, but it is not clear how these statements should be used in the real network systems consisting of many different application security frameworks, firewalls, etc.

It is clear that the bulk part of the Internet users are accessing services through the web browser software installed on their workstation. It could be expected that the web browser will be the primary user interface software also in the future, but the user experience of the Internet should change considerably. It is not possible to maintain different security credentials for a number of Internet sites by the user. Client-side password lists are only a temporal solution to this problem. It is expected that personalizable user-oriented portals will spread broadly across the Internet and that a centralization and better control over the user information will be enforced.

Trusted organizations like banks, telecommunication operators or service providers could play a role of digital identity providers and provide digital identity management services.

It is apparent that several competing identity provides will emerge and that a mechanism must exist to locate the identity provider. Some such mechanisms are proposed as a part of digital identity systems, but these are very simplistic and are limited by the existing Internet browsers capabilities. As the technology will evolve, new mechanisms will be needed for the efficient identity provider location service.

The digital identity systems described in this work do not deal with the web services directly. However web services may become an important part of the Internet applications and should be considered with regard to the user identity. Several unsolved issues can be seen when considering the web services security from the digital identity point of view. Some of these include authorization mechanisms, proxy authentication and authorization of web service access and service accountability issues. The most of these issues are not directly dependent on the web services environment, but in this context are the most apparent. It is clear that those issues must be addressed before digital identity systems could form a base of complete identity infrastructure for the Internet.

6 Thesis objectives

The digital identity technologies will probably evolve and became the base for a next generation security infrastructure for the Internet as well as corporate intranets. Nevertheless there are still some issues that must be resolved before these technologies will be used to their full potential. Several of these issues may be addressed in the future work.

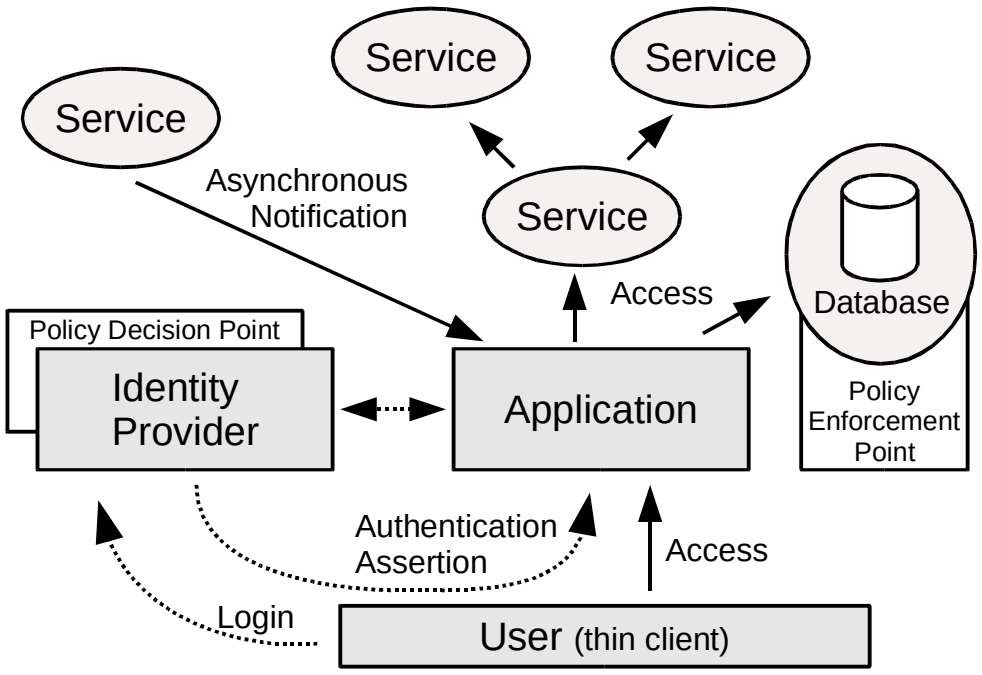

Figure 14 Distributed web system overview

Figure 14 shows the concept of a distributed web application system. We consider a user equipped with web browser (thin client), a web application with HTML-based graphical user interface (GUI) and an identity provider that acts as an authentication authority. Typical web application is a front-end to other services, namely database systems. It may be expected that with the proliferation of the web services the web application will compose several such services into one user interface. For example a portal web application may provide user with information from local database, directory server, corporate information system and several remote services on the same screen. Even the services itself may be composed from other services and may use disparate resources to finish their task.

It is clear that the the application should impersonate the user to the services to finish its tasks. While this situation could be acceptable in the enterprise environment where most of the applications are trusted, it is not a suitable solution for the untrusted Internet environment. A mechanism for a controlled delegation of authentication status and/or authorizations is needed in this situation. In addition to this, there are tasks that should be performed on the user's behalf when he is not logged in. Such tasks may include scheduled processes by the means of cron UNIX

command or processes triggered by asynchronous messages. No suitable security mechanism is proposed for these situations.

The Security Assertion Markup Language (SAML) [35] provides the rough outline of the Policy Decision Point (PDP) and Policy Enforcement Point (PEP) scenario. We believe that this scenario could provide a suitable base for the future development of advanced security services and should be used as a starting point for the further research in this area.

According to the previous points, following thesis objectives are proposed:

- Consider the web application/web services concept and analyze the security considerations of this type of application deployment form the digital identity point of view.
- Propose a mechanism that should provide appropriate security services in this environment. Consider composition of the web services as well as traditional database access and directory services access. Focus on the authentication status transfer and/or authorization delegation in the web environment.
- Design a protocol or modify existing protocols to support proposed mechanism. Focus on standard protocols, especially XML-based protocols (SAML, XACML, XTAML, etc.)
- Verify proposed mechanisms and protocols by implementing relevant parts of the system in the UNIX environment. Consider modifying some existing web applications and web services to support proposed mechanisms as the security services.

Bibliography

[1] Chapman, D.B., Zwicky, E.D.: Building Internet Firewalls, 1995, ISBN 1-56592-124-0

[2] Rescorla, E.: HTTP Over TLS, RFC 2818, 2000

[3] Joshi, J.B.D., Aref, W.G., Ghafoor, A, Spafford E.H.: Security Models for Web-Based Applications. Communications of the ACM, Vol. 44, 2001, pp. 38-44

[4] Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E.: Extensible Markup Language (XML) 1.0, W3C Recomendation, 2000

[5] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., Nielsen, H.: SOAP Version 1.2 Part 1: Messaging Framework, W3C Working Draft, 2002

[6] Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL), W3C Recomendation, 2001

[7] Chow, R., Johnson, T.: Distributed Operating Systems & Algorithms, 1997, ISBN 0201498383

[8] Frier, A., Karlton, P., Kocher, P.: The SSL 3.0 Protocol, Research report, 1996

[9] Information technology - Open Systems Interconection - The Directory: ITU-T Recommendation X.500, 2001

[10] Yeong, Y., Howes, T., Kille, S.: Lightweighted Directory Access Protocol, RFC 1777, 1995

[11] Schneier, B.: Applied Cryptography, Protocols, Algorithms, and Source Code in C, 1996, ISBN 0-471-11709-9

[12] Haller, N.: The S/Key One-Time Password System, ISOC Symposium on Network and Distributed Systems, 1994

[13] Haller, N., Metz, C., Nesser, P., Straw, M.: A One-Time Password System, RFC 2289, 1998

[14] Wiener, I. C.: Sample SecurID Token Emulator with Token Secret Import, http://online.securityfocus.com/archive/1/152525

[15] Menezes, A., Van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography, 1996, ISBN 0849385237

[16] Simpson, W.: PPP Challenge Handshake Authentication Protocol (CHAP), RFC 1994, 1996

[17] Information Technology - Open Systems interconnection ...: ITU-T Recommendation X.509, 2000

[18] Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol, RFC 2560, 1999

[19] Housley, R., Ford, W., Polk, W., Solo, D.: Internet X.509 Public Key Infrastructure Certifica, RFC 2459, 1999

[20] Dierks, T., Allen, C.: The TLS Protocol Version 1.0, RFC 2246, 1999

[21] Hickman, Kipp: The SSL Protocol, Research report, 1995

[22] Wagner, D., Schneier, B.: Analysis of the SSL 3.0 protocol, The Second USENIX Workshop on Electronic Commerce, 1996

[23] Callas, J., Donnerhacke, L., Finney, H., Thayer, R.: OpenPGP Message Format, RFC 2440, 1998

[24] Ramsdell, B.: S/MIME Version 3 Message Specification, RFC 2633, 1999

[25] Adams, C., Cain, P., Pinkas, D., Zuccherato, P.: Internet X.509 Public Key Infrastructure, RFC 3161, 2001

[26] Hallam-Baker, P., et al.: XML Key Management Specification (XKMS 2.0), W3C Working Draft, 2002

[27] Bartel, M.,Boyer, J. , Fox, B. , LaMacchia, B., Simon, E.: XML-Signature Syntax and Processing, W3C Recomendation, 2002

[28] Imamura, T., Dillaway, B., Simon, E., et al.: XML Encryption Syntax and Processing, W3C Candidate Recommendation, 2002

[29] Farrell, S., Housley, R.: An Internet Attribute Certificate Profile for Authorization, Internet draft, 2001

[30] Kohl, J., Neuman, C.: The Kerberos Network Authentication Service (V5), RFC 1510, 1993

[31] Zandbelt, H., Hulsebosch, B.: IDsec: Virtual Identity on the Internet, Internet draft, 2002

[32] Microsoft Passport: Technical White Paper, 2001

[33] Erdos, M., Cantor, S.: Shibboleth Architecture DRAFT v04, http: //middleware.internet2.edu/shibboleth/docs/dr

[34] Hodges, J.: Liberty Architecture Overview, Liberty Alliance Project Specification, 2002

[35] Hallam-Baker, P., Maler, E.: Assertions and Protocol for the OASIS Security Assertion Markup Language, OASIS Standard, 2002

[36] Mishra, P., et al.: Bindings and Profiles for the OASIS Security Assertion Markup Language, OASIS Standard, 2002

[37] Microsoft .NET Passport Security and Privacy Overview: Technical documentation, 2001

[38] Korman, D., Rubin, A.: Risks of the Passport Single Signon Protocol. Computer Networks, volume 33, 2000, pp. 51-58

[39] Slemko, M.: Microsoft Passport to Trouble, http: //alive.znep.com/~marcs/passport/

[40] Rouault, J.: Liberty Bindings and Profiles Specification, Liberty Alliance Project Specification, 2002