
JTAP

JISC Technology Applications Programme

Web Security

Andrew Cormack
University of Wales, Cardiff

Web Security

**Andrew Cormack
University of Wales, Cardiff**

The JISC Technology Applications Programme is an initiative of the Joint Information Systems Committee of the Higher Education Funding Councils.

For more information contact:

Tom Franklin
JTAP Programme Manager
Computer Building
University of Manchester
Manchester
M13 9PL

email: t.franklin@manchester.ac.uk

URL: <http://www.jtap.ac.uk/>

Contents:

1 INTRODUCTION.....	3
2 SECURITY.....	3
2.1 Authentication.....	3
2.2 Authorisation.....	5
2.3 Privacy.....	5
3 WORLD WIDE WEB SECURITY.....	7
3.1 Authentication and the Web.....	7
3.2 Authentication Servers.....	8
3.3 Security Requirements.....	9
3.4 The Web Transaction Security Proposal and SHTTP.....	10
3.5 Netscape's Secure Socket Layer (SSL).....	11
3.6 Summary.....	13
4 OTHER INTERNET SERVICES.....	15
5 PRETTY GOOD PRIVACY (PGP).....	16
6 CHARGING FOR SERVICES.....	17
7 CONCLUSIONS.....	18
8 REFERENCES.....	19
8.1 Cryptography.....	19
8.2 IETF documents.....	19
8.3 Implementations.....	19

1 Introduction

Security was not a priority for the original designers of the internet protocols. Networks were assumed to be either private, isolated and physically secure, or else completely public. This assumption is no longer true because of the growing commercial use of the Internet; a considerable amount of private information is now being sent over public networks.

Cryptographic techniques have been developed to keep this traffic secure, however many of the implementations are proprietary and require the two parties to use the same software. A number of standards have been proposed which would allow different systems to exchange secure data over the Internet; these are being considered by working groups of the Internet Engineering Task Force (IETF).

Each internet service has its own security requirements. This report reviews the security measures which are available and under development for the World Wide Web, and considers how well they meet the requirements of the UK academic network, JANET. There is also a brief comparison with other internet services such as file transfer and electronic mail.

2 Security

The security of a computer network has three aspects: authentication, authorisation and privacy. Different networks and services will place different emphasis on each of the three components - a closed internal network may have no need for privacy, while a public service may only be concerned with authorisation (to prevent access to other, private, services on the same machine). The security of the computers connected to the network is also important since an insecure computer can be used to attack any network to which it has access. However this problem is not specific to networked computers and is already addressed by the various Computer Emergency Response Teams (CERTs) so it will not be discussed further here.

2.1 Authentication

The main purpose of authentication is to verify the identity of the person using a service, though it can also be used to prove the identity of the service to the user. The required level of proof will vary depending on the nature of the service, but will usually be based on one or more of the following

- what you know (for example a username and password)
- what you have (a smartcard or other token)
- what you are (fingerprints, retinal scan etc.)

Most security systems require a combination of at least two of these forms of proof: many tokens require that a password be entered to unlock them. This protects against accidental loss or theft of a token, but cannot prevent deliberate deception where a registered user hands their identity to another person. Users should always be warned against doing this, but expensive biometric techniques may be necessary for the most secure information.

Any authentication method which is transmitted across a public network may be recorded by any third party with access to the network. If this person can later use the recorded transaction as their own 'authentication' then the method is of little value. These 'replay' attacks can be prevented by ensuring that the credentials exchanged across the network are different for every transaction, either by using one-time passwords or by encrypting each transaction with a unique key. A further danger of replay attacks is that, unlike the loss of a physical token, the theft may go unnoticed by the rightful owner.

Methods which prevent replay attacks are known as 'strong authentication' and can be divided into three classes: shared sequence, challenge/response and asymmetric key. In shared sequence methods, the user and the service both have access to a sequence of passwords which are used in turn to authenticate the user. The sequence may be printed as a list or be generated on demand by a dedicated calculator or program. Once used, each password is invalidated so a replay attack is bound to fail. The best known of these methods is Bellcore's S/Key which has been implemented on both hardware tokens and general purpose computers. Time based authentication methods, such as Kerberos and SecurID, may also be classed as shared sequence.

In challenge/response systems the service issues a challenge string, which should be different for every transaction, and the user responds with a combination of the challenge and their own password. The operation used to form the response is a message digest function, designed to make it virtually impossible to reconstruct the password from the response. Replay attacks will fail because the response is only valid for the particular challenge issued. The digest authentication scheme included in version 1.1 of the Hypertext Transfer Protocol uses challenge/response (see later) as do commercial systems such as CryptoCARD and Digital Pathways' SecureNet.

Asymmetric key systems use pairs of numbers with the property that a message encrypted using one of the pair can only be decrypted using the other. If every user has their own pair, one of which is widely publicised (the public key) while the other is only known to them (the private key), then these can be used to authenticate the user. If a service receives a message which decrypts correctly using someone's public key then it can be virtually certain that it was encrypted using the private key. Similarly, a service which encrypts its replies with the user's public key can be confident that they can only be read by use of the corresponding private key. The best known implementation of these methods is PGP, now freely available as software for most platforms. An internet standard, PEM, also includes asymmetric keys but is less widely used.

Any service which requires its users to authenticate themselves must have independent information to check the authentication against. This may be held by the service itself, but is often provided by a separate authentication server. This is particularly convenient where many services are provided to the same users: rather than duplicating the information on every server they all refer to the single copy on the authentication server. If the communication between the service and the authentication server uses a public network, it must be secured to prevent bogus information or users being introduced. Individual services may, if required and if the authentication method permits, keep local copies for a short period to reduce the network load of contacting the authentication server for every transaction.

There are two ways in which an authentication server can operate. In the first, the server acts as a "Certificate Authority" (CA) and guarantees the identity of each user (or service) whose credentials it issues. New users can only be added to the server by an authorised person, who will usually check against other documentation such as a driving license or company registration. Each user can only be authenticated by a single Certificate Authority so if that server and any backups are unavailable for any reason none of its users will be able to authenticate themselves. The alternative model, used by PGP, is to have servers which merely issue credentials, without giving any guarantee. Instead, the proof of identity is encoded into the information which is served. PGP public keys can be signed as genuine by one or more third parties, allowing a 'web of trust' to be built up. For example a student's key might be signed by their lecturer, whose key is in turn signed by the head of department. Since signed keys contain their own proof of identity, they can be placed on one or more authentication servers by anyone.

Within JANET it seems likely that authentication servers would be located at institutions with each one holding information about its local users. This would allow the authentication information to be created and maintained within the institution. If the Certificate Authority model is followed then new users may only be added by responsible individuals appointed by the institution. If the signed key system is used, only the signing process needs to be controlled by the institution. Signed keys can be placed on multiple servers, perhaps at different institutions to increase their availability.

Authentication is mainly used to identify a user to a server, but also has other applications. If a service requires that a user provide sensitive information, such as a credit card number, the user may reasonably require that the service authenticates itself before receiving the information. It may be useful to attach a proof of identity to a document, both to authenticate the author and to prove that the text has not been altered. Successful authentication can be recorded as an electronic signature, providing irrefutable proof of some action.

2.2 Authorisation

Once a user has proved their identity, the service must check whether that person is allowed to perform the operation which they have requested. This authorisation is normally done within the service machine by checking against a list of registered users and their access rights. For some services, however, it may be necessary for the lists of users and their rights to be maintained by the users' own institution, rather than the provider of the service. This might apply where an institution has paid for a license to use the service and needs to control which users have access to it.

A secure protocol is then needed for the service to query an authorisation server at the institution. A single service may have different areas of information, for example different datasets, each of which has a different group of authorised users. This means that the request for authorisation must include details of the information being requested by the user, not just the identity of the service involved.

The simplest approach would be to combine the functions of authorisation and authentication servers at the institution and only issue authentication information to those services for which the user is authorised. However this requires each authentication server acts as a Certificate Authority. It also means that the same authentication information cannot be used for signatures and electronic mail, since requests to validate signatures may come from any host on the Internet.

2.3 Privacy

The Internet carries an increasing amount of private traffic. This may be personal information about the user or information of commercial value. Whether the messages contain credit card details, purchased software or examination marks, their owners need to keep them secret. Unfortunately messages can be read off the network as easily as usernames and passwords so the only solution is to encrypt them. It is important to note that any mathematical encryption scheme can be broken by the use of sufficient computing power: the best that can be hoped for is to make the cost in time and CPU power needed to break the code significantly greater than the value of the encrypted material.

If an authentication process occurs before a user gains access to a service, then it may be possible to use information gained during authentication to encrypt the subsequent traffic. This requires care to ensure that only the intended user can decrypt the traffic, so is most suited to asymmetric key methods. If the server issues a public key as proof of its identity then this key may be used to negotiate an encryption method for the subsequent session. This

may also be done in reverse if the server knows the user's public key. Asymmetric key systems require too much calculation to be used directly to encrypt the whole session; the usual method is to choose a random symmetric key for the session and exchange this securely using the asymmetric keys. If neither of the parties has been authenticated then it is still possible to agree a session key, however this provides no protection against clients or servers obtaining information by deceit.

The internet protocols are based on nesting different layers of information so there is a choice of layer at which to apply encryption. One approach is to encrypt at the Transport Layer, leaving un-encoded only the information required for to route packets to their destination. The network simply transfers the packets between the two endpoints, which are the only machines capable of making any sense of the information. The encryption occurs below the level where different services (WWW, E-mail, FTP, etc.) are distinguished, so can be used equally well by any of them. Transport layer encryption is used by Netscape's Secure Socket Layer (SSL) protocol, also known as TLS, and is proposed as an option for the new version of the Internet Protocol itself, IPv6.

While it seems attractive to apply a single encryption method to all internet services, some services benefit from specific operations performed by intermediate systems. For example mailbagging reduces the bandwidth required to send multiple copies of electronic mail messages to distant sites, FTP requests may be re-directed to local mirror sites and web requests may be serviced by caches rather than the origin servers. Each of these operations requires that some intermediate machine be able to read the request contained within a packet, so is impossible with transport layer encryption. The alternative is to encrypt at the application layer, leaving the useful header information readable but encrypting the content. The most popular system for encryption at this level is Pretty Good Privacy (PGP) which is widely used for e-mail and FTP, and is one of the options supported by the proposed Secure-HTTP (SHTTP).

Secure encryption is regarded with suspicion by governments and many countries have legal limits on its use and distribution. American law restricts the export not just of software which performs encryption, but even of programs which allow encryption to be added after export. Both PGP and SSL are restricted but their algorithms have been re-implemented outside the USA to evade the prohibition. However there remains a risk that future developments in the Internet may rely on technology which is less easily available in the rest of the world. Even IPv6 is likely to suffer since it uses DES encryption which has been subject to restrictions in the past.

3 World Wide Web Security

The World Wide Web was originally developed as a publishing medium for public documents, so provided few controls for restricting access to information. As a wider range of documents and services appeared on the web, these needed improved security facilities and a number of systems were proposed to satisfy the new requirements. This section sets out the security needs of users, publishers and authors on the web, and examines two alternative solutions. However the HTTP protocol presents particular problems for authentication, so this aspect is considered first.

3.1 Authentication and the Web

When using an FTP or telnet service, the user is authenticated during the login process, sends one or more commands to the service, and then logs out. The initial authentication remains in effect for all of the operations performed until the user logs out; the many interactions between user and server during this period are therefore regarded as a single session. The HTTP protocol has no concept of a session. Each connection which a user makes to a server carries only a single request and response and is independent of all other connections, even those between the same two parties. The protocol was designed in this way so that servers did not have to retain any information from one connection to another; in particular this means that authentication information, if required, will not be retained by the server and must be included with every request. So long as the same authentication can be used for all documents on that server (or a known subset of those documents) this is not a problem as the necessary credentials can be stored by the browser program and reissued for each request. Any system which reuses the same credentials is vulnerable to replay attacks but the alternative of using a one-time password token requires that the user enter a new password for every request. Since each displayed page may involve several requests (every graphic on the page requires a separate request) this would soon become intolerable. Systems like S/Key which store a limited number of passwords would rapidly be exhausted.

Version 1.0 of HTTP provided only basic authentication, using a static username and password. When a user requests a protected document, the server replies with the error "Unauthenticated" and an indication of the document subset, or 'realm'. The browser should prompt the user for an appropriate username and password and re-send the request with these credentials included in a header. If the credentials are accepted by the server, it will return the document requested. The browser can later use the same username and password, without consulting the user, in response to other "Unauthenticated" errors from the same server and realm. In version 1.0 of the protocol the credentials are encoded, but not encrypted, in the request so they can easily be recovered by anyone monitoring the network. This person may later use the same username and password to retrieve any document from the same realm, even those which were not accessed by the authorised user.

Version 1.1 of HTTP (published in July 1996) introduces an improved method called digest authentication. This uses the same exchange of packets as basic authentication, but the "Unauthenticated" reply now includes a value, known as the nonce, which acts as a challenge. Instead of replying with the username and password, the client calculates a message digest (using the MD5 algorithm) from the username, password and nonce and returns this along with the username as authentication information. The server then repeats the MD5 calculation, using the user's correct password, and returns the document if the two digests match. To do this the server must hold each user's password in a form suitable for calculating the MD5 digest. It is imperative that these passwords be stored as securely as possible, since anyone possessing them would immediately be able to masquerade as any valid user of the server. The need for security is even greater than for a Unix password file, whose contents must be de-crypted by brute force methods before they can be misused.

The HTTP server is stateless so cannot "remember" the nonce value between each challenge and its response; the nonce must therefore be derived from some combination of information from the request packet along with values held centrally on the server. This still leaves the process vulnerable to replay attacks as the server does not ensure that nonces are unique to a single request. However by careful choice of the nonce this risk and the resulting damage can be substantially reduced. A good nonce calculation will usually include the URL of the document requested so that a successful replay attack can only retrieve a single document, rather than the whole realm as with basic authentication. The IP address of the client making the request should also be part of the calculation: a replay attack from a different client machine will then fail unless the intruder has changed his IP address and managed to persuade the routing systems to return responses to the new location. The server may also maintain a single 'magic number' which changes periodically. Including this in the nonce will limit the time period during which a replay attack can be successful. However the rate of change must be sufficient to allow a legitimate user to enter a username and password as otherwise the nonce will have expired by the time the authenticated request is sent. To assist the user, it is suggested that servers should recognise digests which are correct except for using an 'old' nonce and request that the browser program re-calculate the digest with the latest value. This second calculation can be done without consulting the user (since the browser will have cached the username and password) so should have a shorter round-trip time.

Only the server needs to know how the nonce is calculated, so different servers can choose methods appropriate to the sensitivity of the information they hold. Each improvement in security requires more processing in the calculation of the nonce and the message digest; for a busy server this may represent a considerable load. A nonce with a short lifetime is also likely to require many re-authentications, increasing the network traffic. The only way to prevent replay attacks is to keep a record of all digest values received by the server, and ensure that each one is only used once. However this involves the server holding a great deal of information which must be checked for every request. There is still no protection against other types of attack, for example through a compromised or hostile proxy between the client and the server. These can only be prevented by cryptographic encryption of the request and response so that even if transactions fall into the wrong hands, they are of no use. The standard concludes:

"Digest Authentication does not provide a strong authentication mechanism. That is not its intent. It is intended solely to replace a much weaker and even more dangerous authentication mechanism: Basic Authentication. An important design constraint is that the new authentication scheme be free of patent and other export restrictions.

Most needs for secure HTTP transactions cannot be met by Digest Authentication. For those needs SSL or SHTTP are more appropriate protocols."

SHTTP and SSL are described later in §3.4 and §3.5.

3.2 Authentication Servers

Most current web servers perform their own authentication, using information held within the server. A modified server could instead refer to a remote authentication server when it needs to check a user's credentials. The use of a separate server to provide strong authentication is outside the scope of current web proposals, which deal only with the interaction between the client and the web server.

If there were many authentication servers (for example one for each institution) then the userid would have to be used identify the correct authentication server for that individual. The normal HTTP authorisation protocol (used by both basic and digest methods) transmits the

userid and password together, so an authentication server could only be contacted *after* this exchange. This means that the authentication server cannot provide a challenge string for the user, but must rely on the one issued independently by the web server. Some challenge/response systems can work with this restriction, but others do not. It might be possible to request the username and password by two separate transactions, with a reference to an authentication server between, but this would require modifications to the browser program to avoid presenting a very clumsy interface to the user.

An authentication server may either provide information, such as a public key, which allows the server to perform the authentication or may perform the authentication itself. In the latter case the link between the web server and the authentication server must be secured to prevent useful information being stolen or false authentications being introduced. If the web server does not have to know the authentication method in use, but merely passes on the request and credentials for approval, this makes it much easier to define local authentication methods within the institution. For example an institution's authentication server might control smart-card readers attached to each terminal. Whichever form of authentication server is used, the web server must have some method for sending authentication requests to the server. Most hardware tokens have proprietary servers and use their own protocols. To avoid the need for every secure web server to know which authentication server is used by each institution, a common protocol or gateway needs to be provided by all servers.

A further problem with token-based systems is the lack of recognisable sessions in the web protocol. There is no reliable way for the server to identify a sequence of connections as coming from the "same" person, so it may be necessary to authenticate every request individually. This would result in a very slow system, which would only be acceptable for the most sensitive information, but the alternative of caching authentications on the web server may run the risk of replay attacks. One form of authentication which can safely be cached is asymmetric key encryption. A user's public key can be cached on the web server for a period, so only the first request of a session would be delayed by contacting the authentication server.

3.3 Security Requirements

User authentication is only one aspect of web security. The best possible user authentication and authorisation is of little value to a web server if the restricted document is then transmitted in clear text across the Internet. The reply should therefore be encrypted to prevent the information being read or modified by others. Some web documents may only be read from certain IP addresses; to secure these against attacks on the Domain Name Service (DNS) it may be necessary to perform strong authentication of the client machine to prove its identity, however once strong authentication of the individual user is possible this requirement may diminish. A more subtle form of attack would be to modify an incoming request, either to mislead the user or to gain additional information, so some servers may require that the text of the request be signed so that tampering can be detected.

The user of a web service may have their own security requirements. Some requests, particularly those resulting from filling in forms, contain personal or other secret information. To prevent this being read by individuals monitoring network traffic, it should at least be encrypted. The careful user may also require that the server authenticate itself before receiving such information: even discounting attacks on the DNS, internet names are not subject to trademark legislation and, particularly in the commercial (.com) domain, do not always belong to the "obvious" owner.

There are some documents whose text is public but which need to be certified as genuine. Price lists, press releases and articles in electronic journals are examples. The author of these documents needs to know that modified copies of the document cannot be passed off as

genuine and the reader needs a guarantee of origin before acting on the information. This authentication of authorship is distinct from authentication of the server where the document is held: for many of these documents it does not matter where a copy is obtained from, provided the copy is known to be genuine. The most common way of attaching a "digital signature" to a document is to calculate a message digest value for the document (using MD5 or some other algorithm) and then encrypt this with the author's private key. The encrypted value is appended to the document before publication. The reader of a document may be sure that the text has not been altered if decrypting the signature with the author's public key gives the same value as re-calculating the message digest. The operations of signing and checking are usually carried out off-line; the signed document may be distributed as a single object by any appropriate means (FTP, http, e-mail, floppy disk etc.). Client programs such as web browsers may include signature checking as a convenience, but there is no need to do so.

Some services may also require lasting proof that a transaction took place. For example a supplier may wish to have irrefutable proof that an order was placed, while the customer will require proof that it was accepted. This non-repudiation is required by all commercial services but also has other applications. Strong authentication is usually required to confirm that a transaction took place, with a digital signature to ensure the details cannot be falsified.

These requirements are summarised in §3.6 below.

3.4 The Web Transaction Security Proposal and SHTTP

The Internet Engineering Task Force (IETF) have published a draft document on security called 'Requirements for Web Transaction Security (WTS)'. This identifies the following services which are needed to make the web secure:

- Confidentiality of the HTTP request and/or response
- Data origin authentication and data integrity of the HTTP request and/or response
- Non-repudiability of origin for the request and/or response
- Transmission freshness of request and/or response
- Ease of integration with other features of HTTP
- Support of multiple mechanisms for the above services

This list meets most of the requirements outlined in §3.3 above. Encryption and signing of both request and response are included in WTS (signing ensures "data integrity"), as is authentication of the server. It is not clear whether the "origin" of an HTTP request is considered to be a client machine or the person using it, however elsewhere in the document there is a statement that user authentication is a requirement. The phrase "data origin authentication" is also ambiguous but a later section makes it clear that this refers to authenticating the server, not the author of a document:

"WTS ... does not provide independent certification of documents or other data objects outside the scope of the transfer of said objects."

Author authentication should therefore be achieved, as on other internet services, by the use of digital signatures on each document.

One implementation of the WTS proposals is the Secure Hypertext Transfer Protocol (SHTTP), produced by Enterprise Integration Technologies (EIT) and now a draft internet standard. SHTTP provides a mechanism for browser and server to agree on their security requirements and adds information to the normal HTTP headers to allow signed and

encrypted requests and responses to be sent and received. The basic mechanism is to take a normal HTTP request or response, encrypt and/or sign it as agreed, and then enclose it in an SHTTP request which carries only sufficient information to allow the authorised recipient to decrypt the contents.

A reference implementation of SHTTP was written by modifying the Mosaic browser and NCSA web server. This uses PGP public-key encryption which is available internationally. The hypertext link to a secured document includes the server's public key (or an identifier allowing the key to be obtained from elsewhere) and this is used to encrypt the initial request from the client. This request includes the client's public key and is signed using the client's private key as authentication. Subsequent transactions are encrypted using the recipient's public key, and may be signed with the sender's private key, thereby achieving the necessary cryptographic security.

The reference implementation requires that each user's public key be installed on the server by an administrator who is expected to take appropriate precautions to ensure that the key does belong to the registered user. Signed public keys could instead be obtained from a key server, if required, and could then be retained by the web server.

3.5 Netscape's Secure Socket Layer (SSL)

The WTS paper proposes that web security should be based around the Hypertext Transfer Protocol. This would have no impact on other application protocols, such as FTP, telnet and e-mail, which would be free to adopt their own security measures. Security could be matched exactly to the needs of each application but at the cost of some duplication of effort and perhaps greater complexity for the user.

An alternative model, which has already been implemented for the Web, is to provide security at the lower, transport, layer. This provides an encrypted TCP connection between client and server machines which can be used by any application protocol. Provided the encryption is strong enough, none of the transaction can be read from the network by any third party. This is the basis of the Secure Socket Layer (SSL) protocol developed by Netscape and others and now adopted by the IETF under the name Transport Layer Security (TLS).

SSL is a general-purpose system so it cannot offer services which are tailored to the application which is using it. On the web, requests within SSL channels cannot benefit from caching or the intelligent re-direction which some servers now offer. Special arrangements must be made for SSL channels to pass through firewalls and other proxies; these may also provide routes through the firewall for unwelcome visitors. Since the channel is encrypted there is no way for the firewall to even monitor what is passing through it.

Netscape's implementation of SSL is subject to export restrictions and only a version with reduced strength encryption is available outside the USA. Netscape themselves suggest that with the export version each encryption key should only be used for 24 hours or 500 transactions before being changed. The algorithms are published and an alternative public-domain implementation of full-strength SSL is available from Australia. This has been used to add SSL capability to the source code of other browsers and servers; unfortunately the Netscape browser and server are supplied as executable programs so cannot have their encryption upgraded in this way.

Comparing SSL to the WTS aims, SSL provides confidentiality and authentication of request and response messages. It can be used to exchange certificates to authenticate the server and client machines, however these assume the presence of third party Certificate Authorities which may not be appropriate in the UK. No record is kept of each authentication so non-

repudiation is not possible. The main problems with SSL are the low level of security available in the export version and the difficulty of interacting with application-specific intermediaries such as proxies and caches. Despite its strong commercial support, SSL is not a complete solution to web security.

3.6 Summary

3.6.1 Requirements

The first table indicates the components of security which may be required by web readers and publishers on the JANET network, as discussed in §3.3 above. Some services may be prepared to accept lesser measures, but some will require all of them.

Authentication	strong authentication of server; strong authentication (possibly with choice of authentication method) of user with reference to institution.
Authorisation	user access to service authorised by institution.
Privacy	request and response must be private.

3.6.2 Implementation

The following tables show the security offered by current and proposed systems. The final column in each table indicates whether these satisfy the requirements above.

- 1) A standard browser and server, using HTTP version 1.0. This is the most common situation in the UK at the end of 1996:

Authentication	server not authenticated; user authenticated by static, public, username & password	No
Authorisation	web server holds authorisation for individual users or IP domains	No
Privacy	request and response are public	No

- 2) Netscape Navigator, or other SSL compatible browser, accessing a server authenticated by VeriSign using version 1.0 of the HTTP protocol. This is available now, but with only a reduced strength encryption outside the USA:

Authentication	server authenticated by commercial third party; user authenticated across encrypted channel by static username & password.	Yes for server; no for user
Authorisation	web server holds authorisation for individual users or IP domains.	No
Privacy	request and response encrypted	Yes, but export encryption is weak

- 3) A future system, implementing the Requirements for Web Transaction Security. The WTS document does not specify the method used for authentication or encryption, but states that "WTS must be compatible with multiple methods for authentication and encryption". It is possible that commercial implementations will be subject to export restrictions as above:

Authentication	server and user authenticated by non-repudiable method	Yes
Authorisation	web server holds authorisation for individual users or IP domains.	No
Privacy	request and response encrypted	Yes, provided strong encryption is available

4 Other Internet Services

Many of the issues raised by web security apply to other internet services as well. User authentication, document signing and encryption are common requirements and it seems reasonable to expect that the same tokens or keys should give access to most services.

Remote logins were the original requirement for which one-time passwords were introduced. A wide range of proprietary login systems are available, each with their own tokens implemented in hardware or software. Encryption of the subsequent terminal session is rarer; many systems are designed to protect dial-in lines which are much less likely to be monitored than an internet connection. Authentication alone can often be implemented by simply replacing the login program: the user's client program and the main server need not change. SSL may be used to provide an encrypted login session if both client and server support it; alternatively the Secure Shell (SSH) protocol provides both strong authentication and an encrypted session but also requires dedicated client and server software.

Electronic mail may be signed or encrypted. Although there is an internet standard, Privacy Enhanced Mail (PEM), it is little used; instead the de facto standard for both signing and encryption is Pretty Good Privacy (PGP). Most mailers have some way to incorporate PGP when both sending and receiving mail but few are as smoothly integrated as, for example, SSL is into the Netscape browser. Usenet news also makes wide use of PGP for signing messages - encryption of content is seldom a requirement on this broadcast service.

FTP can act as an authorised service, in which case the security options are similar to those for remote login above, or as an open archive. Public files in FTP archives are often signed using the author's private PGP key to prevent tampering. A file for limited distribution may be encrypted using the public key of each of the intended readers (this does not multiply the size of the encrypted file) or, if the readers are not known at first, the file may be encrypted with a unique key which is then issued to readers as they are authorised.

5 Pretty Good Privacy (PGP)

The most common form of authentication and encryption on the Internet (as opposed to just the World Wide Web) is Pretty Good Privacy. PGP is the established standard for e-mail, Usenet and FTP traffic because it is convenient to use and available at full strength in the public domain both inside and outside the USA. PGP uses a random key to encrypt each transaction using the symmetric IDEA algorithm and encrypts the IDEA key using RSA asymmetric keys. Like other asymmetric key methods it can provide strong authentication, even on transaction-based systems such as the World Wide Web. As the most widely available form of strong encryption, PGP is likely to be the method of choice for any implementation of the Web Transaction Security proposals. Public key encryption is particularly attractive since even if an intruder manages to forge a request to a service they still have the problem of decrypting the response.

To use a PGP-based service, the user must know the public key of the service while the server must know (or be able to obtain) the public key of the user. Signed PGP keys can be freely distributed and copied so there are a number of ways in which this can be achieved. The simplest is for the service to be given the public key of each user at the time they are registered; the user, in turn, receives the public key of the service. Each party keeps a file (or key ring) containing the keys of the individuals or services with whom secure communication is required. The keys are public, so there is normally no need for the key ring file to be private. Since the necessary public keys are available locally at each end of a transaction there is no need for any additional network traffic for authentication.

This model assumes that all parties know in advance who they will be communicating with so that the necessary exchange of keys can be performed in advance. To avoid this requirement, a public key server was developed at the Massachusetts Institute of Technology and the software placed in the public domain. The key server has both e-mail and HTTP interfaces which allow any individual or service to add or retrieve signed keys to the repository. The system also includes scripts for key servers to exchange new keys with one another. A user wishing to contact an authenticated service can first obtain its public key from a key server, check that the signatures on the key are sufficient to ensure that the key is genuine, and then add the key to their own key ring. At present keys are retrieved manually and separately from the secure communication itself. It would also be possible for the browser or service to contact a key server as part of the establishment of communication. This might involve a delay in servicing the first request, so the benefits would have to outweigh this disadvantage. Finally, PGP keys are relatively small, typically a few hundred bytes, so the disk storage required for a service to hold a public key for every user on JANET might not be excessive. New users could be registered with their local key servers with new (and cancelled) keys being passed between key servers and services using background scripts running overnight or at weekends.

PGP keys are certified by being signed. A service would normally require that a chain of signatures and keys could be followed from each individual user to a person who is known, and trusted, by the service. The reverse chain linking the service key to the user is also required to give the user confidence in the authenticity of the service. In an academic context it is likely that there would be a hierarchy of signed keys leading to and from a responsible person appointed by the institution. These arrangements for key signing are an administrative, rather than technical, problem.

6 Charging for Services

At present most web services are available free of charge, since there is no sure way either to identify the individual who is accessing the service or to ensure that no others read the information as it passes over the public network. Many organisations which are reluctant to give their product away would be much more likely to make material available on the Internet to paying subscribers. Academic users might well be prepared to pay a subscription or license fee for access to electronic publications, commercial databases or indexing services.

The WTS proposals, described above, provide both confidentiality of response and non-repudiability of request. When combined, these give a publisher the necessary assurances that the information will only be accessible to subscribers and that the subscribers can be identified and their activities recorded. Such a record can provide the basis for an invoice to each subscriber (or their institution) based either on the documents read, or as a per-user or per-site license fee.

This arrangement, where subscriber institutions have accounts with the publishers of information, seems most appropriate to the JANET network. Alternative proposals, collectively known as electronic cash (e-cash), will allow for charging at the time of delivery of a service. However one of the main aims of e-cash is to prevent the 'bank' from knowing the nature of the service being purchased. Its role is simply to authorise and enact the transfer of funds from the purchaser to the vendor. On JANET the user's institution would take the role of bank, but would also expect to know (as authorisation server) what service is being provided.

7 Conclusions

The existing web security mechanisms have been developed to support shopping on the Internet. Before sending their credit card number over the network a user needs to be confident that the number will not be misused by the recipient and cannot fall into other hands as a result of network monitoring. In security terms this requires that the remote server be authenticated and that the communication be encrypted. The goods which are purchased are not usually delivered over the network but by traditional postal services. These have their own ways of ensuring that delivery is made to the correct person (in other words authenticating the recipient), so there is no need for the user to be authenticated during the network part of the transaction. Netscape's SSL and VeriSign provide exactly the required facilities: server authentication and privacy.

On an academic network transactions are more likely to involve supplying electronic goods or information. In this case the network *is* responsible for ensuring that the goods are delivered to the correct person. This requires user authentication and an irrefutable record of the transaction. If the information is free, or payment is made off-line by license fee or through an institution's account with a service, then there is less need for the service to be authenticated at the time of the network transaction since this can be done later when the account is presented for payment. Privacy of communication is still needed, but this time to prevent the information (rather than the payment) being read by a third party.

Current web security systems do not address these needs for strong authentication of the user and non-repudiation. Furthermore, although the SSL protocol can support strong encryption, this is only available in a weakened form outside the USA. The encryption which is provided by export versions of Netscape and other commercial browsers is not adequate for sensitive material, whether credit card details, examination marks or other personal or commercial information.

These limitations are addressed by the IETF's Requirements for Web Transaction Security which, if implemented in full and with full strength encryption, would meet all the security requirements of the JANET network. The WTS proposal includes strong authentication of both user and service, non-repudiation and encryption of all traffic. The principal implementation of the proposal, SHTTP, followed the Internet trend by using PGP both for strong authentication of user and service and to encrypt the communication. SHTTP is now a draft Internet standard so seems the proposal most likely to be adopted by commercial suppliers of browsers and servers. Unfortunately there is a risk that these companies will remain committed to their own proprietary security systems. If this happens then a secure web, accessible from any hardware and software platform, will be extremely hard to develop.

Asymmetric key systems such as PGP offer the most practical way to authenticate users of web services since traditional one-time password tokens are not suited to the transaction-based HTTP protocol. They also have the advantage that the authentication information (the public key) can safely be cached by users or services. This makes it possible to distribute relevant keys in advance of their use, removing the need for the web server to contact an authentication server as part of the process of issuing every document. Software to publish and distribute batches of PGP keys already exists and could be used to transfer public keys from their issuing institutions to the appropriate services. Alternatively a few central servers could be set up to hold authentication keys for all JANET users. Such an infrastructure would also be useful for other internet services - such as FTP, e-mail and news - which already make extensive use of PGP signing and encryption.

If the service can safely perform authentication on its own then it seems unnecessary to handicap the system by requiring reference to a remote server for authorisation. Most services

will continue to hold their own lists of access rights for individual users, granted at the time of registration. Remote authorisation could be implemented, if necessary, by passing all requests through a script on the web server but this could lead to an excessive delay in retrieving a document. A secure authorisation protocol would have to be designed, as this minority application seems unlikely to be addressed by any Internet standard.

8 References

The standard textbooks are B.Schneier, *Applied Cryptography*, for details of almost every cryptographic system and S.Garfinkel, *PGP: Pretty Good Privacy*, for PGP in particular. Other documents available in electronic form are listed below.

8.1 Cryptography

S/Key - <http://www.bellcore.com/SECURITY/skey.html>

Kerberos - <http://web.mit.edu/kerberos/www/>

SecurID - <http://www.securid.com/Security/tokens.html>

CRYPTOCARD - <http://www.cryptocard.com/rb1.html>

Digital Pathways - http://www.digipath.com/sec_keys.html

PGP - <http://www.ifi.uio.no/pgp/>

PEM - RFCs 1421-1424 at <http://ds.internic.net/rfc/>

8.2 IETF documents

HTTP v1.0 - <http://ds.internic.net/rfc/rfc1945.txt>

HTTP v1.1 - <http://www.w3.org/pub/WWW/Protocols/Specs.html#HTTP1.1>

Digest Authentication - <http://www.w3.org/pub/WWW/Protocols/Specs.html#Digest>

Web Transaction security requirements - <ftp://ftp.ietf.org/internet-drafts/draft-ietf-wts-requirements-01.txt>

SHTTP - <ftp://ftp.ietf.org/internet-drafts/draft-ietf-wts-shttp-03.txt>

8.3 Implementations

SHTTP - <http://www.eit.com/creations/s-http/>

SSL - <http://home.netscape.com/newsref/std/SSL.html>

SSLey - <http://psych.psy.uq.oz.au/~ftp/Crypto/>

VeriSign - <http://www.verisign.com/>