Data Certification Methods and Apparatus

This invention generally relates methods, computer program code, data processing apparatus, and signals for certifying data, in particular by means of an electronic signature. Embodiments of the invention can be implemented on a user terminal without the need for dedicated hardware or software and may be termed "zero-footprint" data certification methods.

With the increasing use of computer networks such as the Internet for business and other transactions there is a growing need for improved techniques for digitally signing data.

One commonly used technique for generating digital signatures employs asymmetric or public key cryptography which uses a pair of keys, one private and one public (although in practice distribution of the public key is also often restricted). A message encrypted with the public key can only be decrypted with the private key, and vice-versa. An individual can thus encrypt data using the private key for decryption by any one with the corresponding public key and, similarly, anyone with the public key can securely send data to the individual by encrypting it with the public key safe in the knowledge that only the private key can be used to decrypt the data.

Asymmetric cryptography was first publicly disclosed by Diffie and Hellman in 1976 (W. Diffie and D.E. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, 22 (1976), 644-654) and a number of asymmetric cryptographic techniques are now in the public domain of which the best known is the RSA (Rivest, Shamir and Adleman) algorithm (R.L. Rivest, A. Shamir and L.M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", Communications of the ACM, 21 (1978), 120-126). Other more recent algorithms including elliptic curve crypto systems (see, for example, X9.63, "Public key

cryptography for the financial services industry: Key agreement and key transport using elliptic curve cryptography", Draft ANSI X9F1, October (1999)).

An infrastructure known as PKI (Public Key Infrastructure) is often used to provide key management functions. A Public Key Infrastructure normally includes provision for digital identity Certificates. To prevent an individual posing as somebody else an individual may prove his identity to a certification authority which then issues a certificate signed using the authority's private key and including the name and public key of the individual. The Certification Authority's (CA's) public key is widely known and therefore trusted and since the certificate could only have been encrypted using the authority's private key, the public key of the individual is verified by the certificate. Further details of PKI can be found in the X.509 specifications (PKIX) which can be found at www.ietf.org, all hereby incorporated by reference.

Electronic signatures, or MAC (message authentication code) values may also be generated using symmetric cryptography, for example computed using a shared secret key. A well known example of symmetric cryptography is the US Data Encryption Standard (DES) algorithm and its variant triple DES (in which two or three keys are used in succession for additional security); other examples include RC4 and the International Data Encryption Algorithm (IDEA). An example MAC is provided in IS08731-1, "Banking-proved Algorithms for Message Authentication-Part I; DEA"; another example is described in "Computer Data Authentication", National Bureau of Standards, FIPS113, 1985.

A digital signature may be generated by encrypting either the message or a digest or hash of the message. Providing the recipient has the original message they can compute the same digest or hash and thus authenticate the signature. Examples of one-way collision-resistant (hard to guess) hash functions are given in Rivest, "The MD4 message-digest algorithm," Internet Request for Comments 1320, April 1992, and R. Rivest, "The MD5 message-digest algorithm," Internet Request for Comments 1321, April 1992.

The signatures described in this document may be generated using either symmetric or asymmetric cryptography.

Both the symmetric and asymmetric digital signatures described above have disadvantages, in particular requiring the storage of a secret or private key at a user terminal, which may compromise security and create a significant management overhead. For a period a popular alternative was to store the secret key on a chip or smart card but a lack of standardization and high deployment costs have hampered the progress of this approach. The applicant's have previously described, in WO 03/015370, the use of separate signature and authentication servers to provide one solution to this problem. However an architecture which did not require any additional specialised functionality at a user terminal for the purposes of signature creation would be a benefit.

At this point it is useful to review a typical distributed application architecture such as a web application architecture.

Referring to figure 1, this shows a commonly used distributed application architecture in which a user of a terminal or client computer system 10 communicates with an application server 20 via a proxy server 30. Typically communication with the proxy server 30 is over the internet and application server 20 comprises a web server. In practice there may be multiple users and multiple applications. There may be a firewall (not shown) between the proxy server and application server, or between the user and the proxy server.

In the systems to be described below the proxy server 30 is a reverse proxy, that is it represents the content of application server 20 to outside clients, typically providing increased security by preventing direct, unmonitored access to the application server. To the user or client terminal 10 the reverse proxy appears to be the real content server.

In Figure 1 the user has a connection to the proxy, and the proxy a connection to the application. Information is passed between the user and the application via the proxy. Often, the connection from user to proxy or proxy to application is secured (encrypted

and/or authenticated). Authentication may take place in both directions on each connection, i.e. from proxy to user workstation and user workstation to proxy.

The proxy may have one or more of a number of functions, for example:

1. Protecting the application from attackers.
2. Providing authentication to the user on behalf of the application.
3. Managing authentication of users on behalf of the application.

A typical example is a web application. Here, the user has a web browser which connects over the internet to the reverse proxy and the application server is a web server, which sits 'behind' the reverse proxy. (There may in addition be other application and database layers behind the web server, but they are not relevant to this discussion). In this case the reverse proxy typically authenticates to the user by means of a cryptographic key pair and a corresponding X.509 certificate, and the user may authenticate himself to the reverse proxy by a variety of means, such as a username and password, token-generated one-time-password or some other token. The SSL (secure sockets layer) protocol may be used for one or more secure connections to the proxy server 30.

It would be of benefit to be able to provide a signing capability for a user in the distributed application architecture of figure 1 without the need for the user to locally store a secret key and moreover without the need for any dedicated software running on the user's terminal, be it pre-installed or downloaded as an applet or similar.

According to a first aspect of the present invention there is therefore provided a method of providing an electronic signature to a server, the method comprising: receiving data for said server at a proxy system for said server; reading said received data to identify a signature request; obtaining a signature for a portion of data associated with said request responsive to said request, and providing said signature from said proxy system to said server.

The proxy system may obtain the signature either by itself signing the data associated with the request (or a hash or digest thereof) or by sending the data to be signed (or a

hash or digest thereof) to a signature server. Likewise the sender (who may be the originator) of the received data may be authenticated by the proxy system, for example by requesting identification data and associated password, or by a separate authentication server, on request from the proxy system.

The data to be signed may originate from a user of the server, for example the user filing in a form for signature, or the data for signature may be provided by the server or proxy system. For example, the data to be signed may comprise a licence agreement or a set of terms and conditions which are provided to the user; in this case it will be appreciated that there is no need for this data to be sent back to the proxy since the proxy system can keep a record of the data sent to the user for signature. Similarly the data to be signed may comprise an amalgamation of data communicated in a plurality of batches. For example, data for a large number of transactions may be provided by, say, the server to the user, possibly over a period of time, and at some later stage the user may be presented with a summary for signature.

Generally there will be a plurality of users/originators, in which case the proxy system may receive or obtains identification of a user or user group and obtains a corresponding signature.

The signature may employ symmetric or asymmetric cryptography (or both). Generally the signature and the data which has been signed are both provided to the server (although a signature allowing message recovery may be used) but in some cases the signature may be additionally or alternatively provided to a second server such as an audit system. There may be a plurality of servers associated with the proxy system, in which case the signature (and associated signed data) is provided to the particular application server being proxied to the user requesting the signature.

It is envisaged that in embodiments of the system the proxy server and one or more associated application servers will be managed by a single entity. Thus preferably the method includes authenticating the proxy system to the originator of the data to be signed, that is to the user terminal or client, the reverse proxy then becoming a trusted entity.

Optionally following receipt of a request for a signature the data to be signed may be returned to user for confirmation of the signing; this may be included as part of the originator authentication procedure.

The signature request may comprise a signature request tag (which may be an address of a signature server) which is attached to a request for data to be signed sent from the (application) server via the proxy server to the client so that this tag requesting signature is automatically returned to the proxy end in association with the data to be signed provided by the user in response to the request.

In a related method the invention provides a method of providing an electronic signature to a server, the method comprising: receiving data to be signed from said server at a proxy system; sending said data to be signed from said proxy system to a user; receiving a signature request for said data to be signed at said proxy system; obtaining a signature for said data to be signed responsive to said signature request, and providing said signature from said proxy system to said server.

Thus the invention also provides a data carrier carrying a computer network signal, the signal comprising a signature request tag in association with data for signature. The signature request tag may comprise a flag or, for example, a URL or other address of a system for providing a signature. For example, on a form such as an HTML form an action such as a "submit" button may have associated with it an address of a signature server which is part of or associated with the proxy system. In another example the tag may comprise data which is "hidden" that is not displayed to the user but which is nonetheless returned with data captured from the user, for example a hidden field in an HTML form called *signature* and having a null value. In this case the proxy system may provide a value for the signature (optionally as described below in conjunction with a signature server) prior to forwarding the signature (generally with the signed data) to the server. In a related aspect the invention provides a data carrier carrying a computer network signal, the signal comprising a signature request tag in association with an identifier for data to be signed where, for example, the actual data to be signed

has previously been provided (see, for example, the illustration given above of signing a licence agreement).

In a further related aspect the invention provides a method of supplying electronic data to a remote device for signing, the method comprising: receiving data at a user terminal from a first remote device, said received data incorporating a request for user input data and signature request data; inputting user data in accordance with said request; and outputting to a second remote device data incorporating said user input data and said signature request data.

In a further related aspect the invention also provides a method of authenticating electronic data supplied by a user of a source device to a remote device, the method comprising: receiving input data including data for authentication from the source device; identifying authentication request information in the input data; obtaining authentication data for said data for authentication responsive to said identifying, and outputting the authentication data for said remote device; wherein the authentication data authenticates that the supplier of the data is the user.

The invention further provides apparatus to implement the above described methods.

Thus in another aspect the invention provides a data processing apparatus for processing electronic data supplied by a user of a source device to be signed, the apparatus comprising; a communication interface for inputting and outputting data; a data memory operable to store data to be processed; an instruction memory storing processor implementable instructions; and a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to: receive input data including data for authentication from a source device; identifying authentication request information in the input data; obtain authentication data for said data for authentication; and output the authentication data for a recipient device; wherein the authentication data authenticates that the supplier of the data is the user.

The invention further provides a processing apparatus for processing electronic data to be signed, the apparatus comprising: a communication interface for inputting and outputting data; a data memory operable to store data to be processed; an instruction memory storing processor implementable instructions; and a processor operable to read and process the stored data in accordance with the instructions in the instruction memory; and wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to: receive from a first remote device data incorporating a request for user input data and signature request data; input user data in accordance with said request; and output to a second remote device data incorporating said user input data and said signature request data.

The invention further provides a reverse proxy device, the device comprising means to authenticate an application to a user of a user terminal; means to obtain an authentication for said user of said user terminal; means to obtain a signature for user data; and means to receive from said terminal user data for signature and a signature request, and responsive to said signature request, to obtain a signature for said user data and forward said signature and said user data to said application.

The invention also provides a signal, the signal including data for submission to a computer program requiring signed data, characterised in that the signal further includes a request for the data to be signed.

It is also desirable to provide additional protection against some types of man-in-the-middle attacks, particularly those where an internal resource has been tampered with.

Thus, the invention further provides a method of protecting a link between a user terminal and an application server via a proxy system for said application server, the method comprising: receiving data from said application server at said proxy system; determining whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and forwarding said received data to said user terminal responsive to said determining.

Embodiments of this invention allow an internal proxy to check outgoing data, for example to check that all URIs (including URLs) are either on an approved list of machines, and/or an approved list of URIs/paths, and/or internal rather than external (to an organisation, user group or network). This facilitates ensuring that contents of forms previously served have not been tampered with.

The method may comprise inhibiting the forwarding of data from the application server to the user terminal if the data includes data for establishing a connection between the user terminal and a forbidden address. This might appear to the user for example as a connection timeout or a blank document. Alternatively the method may comprise modifying the data to remove data for establishing a connection between the user terminal and a forbidden address. In this example the user might still receive some data sent from the application server but some part of the data would be missing. This part of the data might comprise data normally visible to the user or data normally hidden from the user. Forbidden addresses might comprise any addresses connectable to the user terminal other than via the proxy system. Alternatively, the proxy system may comprise a list of permitted addresses, which are compared with the data sent from the application server. Any addresses found in the data, which are not on the list, would be deemed forbidden. In a further alternative example, the proxy system may comprise a list of forbidden addresses, which are compared with the data sent from the application server. Any addresses found in the data, which are on the list, would be deemed forbidden. In either list example, the list may comprise a list of one or more regular expressions, so that a plurality of addresses may be matched using each regular expression.

In one example the link between the user terminal and the application server via the proxy system comprises a packet data link. Packet data comprises packet header data and packet payload data. The data received from the application server at the proxy system comprises packet payload data. The proxy system determines whether the packet payload data includes data for establishing a connection between the user terminal and a forbidden address. Preferably the link between the user terminal and the proxy system is secure, for example using SSL.

Preferably the determining step comprises parsing the received data for a connection address. The received data may be in a variety of different formats, for example HTML, XML, WML, Adobe™ PDF or a word processor document format. Parsing may comprise parsing data normally visible to the user for a connection address or data normally hidden from the user such as control codes or mark-up tags.

Preferably the proxy system outputs audit data to an audit log if the received data includes data for establishing a connection between the user terminal and a forbidden address. Preferably the proxy system also outputs system management data to provide an alert if the received data includes data for establishing a connection between the user terminal and a forbidden address.

The invention further provides a data processing apparatus for protecting a link between a user terminal and an application server via a proxy system for said application server: receiving data from said application server at said proxy system; the apparatus comprising; a communication interface for inputting and outputting data; a data memory operable to store data to processed; an instruction memory storing processor implementable instructions; and a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to; whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and said received data to said user terminal responsive to said determining.

In one example the data processing apparatus is configured to operate as a reverse proxy server. In addition to reverse proxy functions it provides a means for protecting the link between the user terminal and the application server.

The invention further provides computer program code, in particular on a carrier, to implement the above described methods. The carrier may comprise a data carrier such as a disk, CD-or-DVD ROM, programmed memory such as read-only memory (firmware), or an optical or electrical signal carrier. Code to implement embodiments

of the invention may comprise code in a conventional programming language such as C, or lower level code such as assembly language code, or still lower level code such as microcode or code for setting up or controlling an ASIC or FPGA, or hardware description language code such as Verilog (trademark), VHDL (very high speed integrated hardware description language) or the like. The skilled person will appreciate such code (and/or data) may be distributed between a plurality of coupled components in communication with one another, for example on a network.

The above described feature and aspects of the invention may be combined in any permutation.

These and other aspects of the invention will now be described, by way of example only with reference to the accompanying figures in which:

Figure 1 shows a typical distributed application architecture;

Figures 2a and 2b show, respectively, first and second distributed computer systems embodying aspects of the present invention;

Figure 3 shows a diagram of a computer system including a reverse proxy signature system according to an embodiment of an aspect of the present invention;

Figure 4 shows signalling illustrating a procedure for implementing an embodiment of an aspect of the present invention; and

Figure 5a shows a man-in-the-middle attack preventable by an aspect of the present invention and Figure 5b shows signalling illustrating the man-in-the-middle attack.

Referring now to Figure 2a, this shows how the architecture of Figure 1 may be extended to support digital signatures. In Figure 2a the functionality provided by proxy 40 has been extended to include the role of signature server. The proxy / signature server maintains ownership of the user's private key and uses it to append signatures to data sent from the user to the application, as required.

To indicate that the user wishes a piece of data to be signed, (s)he simply includes some form of 'sign-me' label on that data for this purpose. The proxy / signature server scans all data received from the user looking for such a label. If found, the proxy / signature server creates a signature on the data concerned and sends this together with the original data to the application server.

Before creating such a signature, it is important that the proxy / signature server authenticates the user, so that the correct private key can be used and to prevent use of the key by anyone else. This can be achieved, for example, by requiring that the connection from user to proxy is authenticated by the user; for this any conventional means may be employed.

The application server may have previously requested that the user provide a signed response, so that the user (or terminal) knows when to use the 'sign-me' label. This process may be "automated", with the user's terminal employing the 'sign-me' label on receipt of an instruction to do so from the application server without the need for user intervention.

A number of variations of this basic architecture are possible.

The proxy/signature server described has four main areas of functionality: Acting as a proxy between user and application server; authenticating the application to the user; authenticating the user's identity; and creating signatures on behalf of the user. Some of these functions may be achieved by the involvement of an additional dedicated server, and in particular the latter two of the four functions listed may be delegated to another server.

For delegated authentication the user's identity and/or authenticating data may be stored in an external system. The architecture described with reference to Figure 2a may therefore be modified so that the task of identifying and authenticating the user is delegated to a separate identification and authorisation server.

The task of creating signatures on behalf of the user may also be delegated to an external signature(s) server, rather than having the signature server functionality incorporated directly in the proxy. In this case, the user's authenticated identity is passed to the signature server.

An electronic signature is typically computed based on a representative of the data being signed (a message hash or digest), rather that directly on the data itself. The computation of the message hash may be performed either by the proxy (in which case only the message hash need be passed to the signature server) or by the signature server (in which case the full data to be signed is passed to the signature server).
Both the authentication and signature functions may be delegated, in which case they may be delegated either to two separate servers (one for each function) or to a single combined authentication and signature server.

In any of these cases, a trusted relationship between the proxy and the authorisation server and/or proxy and signature server is required. Mechanisms can be used to ensure the integrity and authenticity of data passed between servers, for example relying on physical security (the servers all being located within a single, secure building) and/or electronic security (secure, encrypted connections being established between the servers).

Figure 2b illustrates an example architecture in which reverse proxy server 40 is coupled both to a user authentication server 40a and to a signature server 40b.

So far, we have considered the case in which the user is authenticated when the connection from user to proxy is first formed, but there are alternatives to this. For example the authentication may take place at any time between the initialisation of the connection between user and proxy / signature server and the receipt of the 'sign-me' label.

User authentication may also take place on receipt of the 'sign-me' label. In this case, the proxy / signatures server requests authentication from the user on receipt of the 'sign-me' label, and proceeds with the signature creation and forwards the signed data

to the application server only once this authentication is complete. This form of authentication may be extended to include user confirmation of the data to be signed. On receipt of data accompanied by a 'sign-me' label, the proxy / signature server returns the data (or summary thereof) to the user, together with a request for authentication. The authentication provided by the user therefore acts as a confirmation that the data presented to the user is correct.

We have considered the case in which the proxy creates signatures on behalf of users, using a different private key for each user. It is also possible for the proxy to use a single key for all users or a group of users, and then to also include the user's identity in the data being signed. In this case the application validates the proxy / signature server's signature, rather than a signature specific to a particular user, which may be a simpler task since a public key specific to the user concerned is not required. The proxy / signature server is trusted to place the correct user's identity in the signed message, but this is no greater level of trust than that bestowed upon it in the previously described scenarios.

Embodiments of the invention described above provide a number of advantages. These include a "zero footprint" – that is the user's terminal is simply required to create a secure connection to the proxy and there is no need to add additional functionality to the terminal for the purposes of signature creation. Integration with the application server is simple – the server must simply put 'sign-me' request labels on data it wishes to be signed, and process the resulting signatures accordingly. Furthermore there is a separation of the signature functionality from the application: Security is achieved by means of administrative and procedural controls at the proxy (and associated authentication and signature servers, if used). Thus the application is restricted from subverting the signature process, particularly where user confirmation of the data to be signed is employed. The cost of employing such controls at the application can therefore be avoided.

Referring now to figure 3, this shows a block diagram illustrating details of the systems of figures 2a and 2b. In figure 3 user or client terminal 10 is coupled to the reverse proxy server 40 by means of a communications network 100 which may comprise any

conventional wired and/or wireless communications network including, but not limited to, the Internet, a wide, local or personal area computer network, a mobile communications network such as a mobile phone network or any other type of network or combination of these.

The reverse proxy 40 comprises one or more communications interfaces 402a, b for interfacing with communications network 100 and one or more web applications 20 provided by one or more corresponding application servers. The communications interfaces are coupled to an internal bus 400 to which are also coupled a processor 404, working memory 406, tamper resistant hardware 408 storing one or more private user keys (optional where a separate signature server 40b is employed), and permanent program memory 410, the contents of which may be provided on a data carrier illustratively shown by disk 412. An example of suitable tamper resistant hardware is the IBM 4758 Hardware Security Module which provides tamper resistant storage and processing. Program memory 410, in embodiments, stores proxy services code, signature request identification code, user authorisation code (optional where a separate authentication server 40a is employed), user certification code (optional where a separate signature server is employed), and operating system code. Optionally program memory 410 may further include HTML parsing and URL blocking code as described in more detail later. Processor 404 loads and implements this code to provide corresponding functions as described further below.

In embodiments a request for user data to be signed together with a request for a user signature, illustratively shown in figure 3 by HTTP form 104 which includes a signature request tag. This issued by a web application 20 and forwarded by reverse proxy 40 over communications network 100 to clients terminal 10 where the user fills in the form and sends data back over network 100 to reverse proxy 40 which recognises signature request tag and signs the user data accordingly before passing the signed data back to the issuing web application.

Referring now to figure 4, this shows signals exchanged by the entities of figure 3 during this procedure. We assume, at the point at which figure 4 begins, that reverse proxy 40 has authenticated itself to a user of client terminal 10, for example by means

of SSL. Thus in web-based system operating within a PKI infrastructure a web browser on client terminal 10 will normally store a root certificate from a certification authority such as Verisign ™ ; such a facility is generally provided as standard even on hand-held mobile devices such as PDAs.

Referring now to figure 4, at step 102 user terminal 10 issues a URL (Uniform Resource Locator) request (input by a user) for a web application 20, which is forwarded by reverse proxy server 40 to the web application server. In response the web application server sends (in this example) an HTTP form including " a sign me" tag to reverse proxy 40 which in turn forwards the form 104 to terminal 10. Terminal 10 receives the form and displays corresponding content to the user requesting input of data including data to be signed. The user inputs this data and the form data is returned 106 to the reverse proxy server 40 together with the "sign me" tag. The reverse proxy server reads the received data, identifies the "sign me" tag and at this stage (in this example) identifies that the user is not authenticated. Proxy server therefore creates a login page which it sends 108 back to terminal 10 where the user inputs a user name and password which is returned 110 to the proxy server 40; alternatively login page may be provided by a separate server.

The proxy server 40 then checks the password by providing 112 the user name and password to authentication server 40a and receiving back 114 from the authentication server user authentication data; alternatively authentication may be performed by the signature server or the reverse proxy server itself. Then, optionally, the reverse proxy server sends 116 the data for signature back to the client terminal 10 together with a request for confirmation that this data is to be signed, receiving back 118 signature confirmation data. Where steps 116 and 118 are implemented obtaining a signature is conditional upon user confirmation. In variants of the procedure user confirmation may be requested at an earlier stage, for example before or at the same time as user authentication.

The reverse proxy server 40 then sends 120 the data to be signed (or a hash or digest of this data) to the signature server 40b together with the user name or other user identification (so that the signature server can select the signature to use), and if

necessary together with a command requesting signature. The signature server 40b finds the data it receives from the reverse proxy server and returns 122 the signature. The signature is then forwarded at 124 by the reverse proxy server to the web application server together with the signed data; alternatively the signature and signed data may be send elsewhere, for example to an audit system. Finally the web applications server sends an acknowledgement signal 126 back to the client terminal 10 via the reverse proxy server 40.

We next describe another implementation of a (reverse) proxy for content and form protection. This may be used together with, or separately from, the above described systems.

Thus optionally the proxy server 40 may also parse HTML data sent from the application server 20 to the user terminal 10 and block unauthorised URLs to prevent some types of man-in-the-middle attacks. Figure 5a shows an example of a man-in-the-middle attack preventable by this method and Figure 5b shows signalling used in the attack. The web application server 20 has been compromised in this example and has been configured to serve web page data incorporating the address of a remote server 50 used by the attacker to collect confidential information from users. The user terminal 10 issues 1 a URL request for a web application, which is forwarded by reverse proxy server 40 to the web application server. This might happen, for example, when the user requests a form. In response the web application server sends 2 an HTML form including a URL to divert the user to the attacker's remote server (using, for example, a URL of the cracker site as the "submit to" field, rather than the application server's own URL), as well as form fields for information such as credit card details, address, etc. The user fills in these details and clicks submit, thinking that (s)he is sending them to the application server. In fact they are sent 3 to the attacker's remote server, and the attacker may collect this information and/or change the values (for example to transfer money to a different account) before submitting it 4 to the application server via the reverse proxy server. It may be impossible for the reverse proxy server to determine that this has happened, since the attacker may spoof the address of the user terminal so that the data appears to have come from the user.

To prevent this kind of attack, the proxy server may incorporate HTML parsing code to parse each HTML document sent from the application server to the user terminal looking for URLs or IP addresses; optionally URL blocking code may also be included (see program memory 410 of Figure 3). This stops the attack method at step 2, before the HTML form reaches the user terminal. The user requests a form from the web server. The web server sends the user the form via the proxy. The proxy checks the form for prohibited URLs and finds one in the "submit to" field. The proxy then logs that this has happened, for example, either by outputting to the standard message output or by writing to a separate audit log. Additionally it could send an alert, for example by generating an SNMP event, which would notify the system administrator that a prohibited URL had been detected. It might also send an error message to the user. An additional refinement would be for the (reverse) proxy, as well as or instead of checking URIs, to put a MAC or other hash of the form in as a hidden field. Other information might include the IP address to which the information is being sent. It can then look to see, when returned forms come back, whether any data has been changed, and whether the returning IP address is the same as that to which it was sent. In all cases, this puts security issues in one domain, and separates them from the application domain.

Any URLs the proxy finds are checked against a list stored on the proxy server, either to allow only URLs on the list or alternatively to prohibit any URLs on the list. If the proxy server finds a prohibited URL, it may either block the HTML document completely or remove the prohibited URL before sending the document on to the user terminal. Since there may be a plurality of application servers connected to the proxy server, it may be advantageous to allow URLs of other application servers connected to the proxy server. This may be achieved by a static list of permitted URLs as already described, or by incorporating part of the routing table for the proxy server in the list, which would have the advantage of being updated continuously by the proxy server's network software. Futhermore, in addition to text string URLs, regular expressions may be used. This allows the use of wildcards to describe a plurality of URLs or IP addresses using a single list entry, and can reduce the length of the list as well as speed up the URL checking process.

The parsing code need not be limited to HTML documents. Since the data sent from the application server to the user terminal may comprise other document formats such as XML, WML, Adobe™ PDF or a word processor document format such as Microsoft™ Word, each having URL-embedding capabilities, a plug-in architecture for different document formats may be utilised. The proxy server identifies the document format from the data received from the application server and runs the appropriate parser code to look for URL data. Furthermore, some document formats allow scripting languages to be embedded within a document, for example Javascript within HTML documents or macros within Word documents. These present an additional challenge as it may not be possible for the parser to determine the result of a script incorporated in a document. Such a script may, for example, generate a forbidden URL when run at the user terminal without incorporating the text of the URL in the document as received from the application server. In this case the parser may block all scripting language code, or permit only certain pre-determined scripting language code passages in the data received from the application server. This works in a similar way to parsing a list of permitted URLs; in addition care must be taken to ensure that it is not possible to generate a forbidden URL using only permitted scripting language code passages in the document.

No doubt many other effective alternatives will occur to the skilled person. It will be understood that the invention is limited to the described embodiments and encompasses modifications apparent to those skilled in the art lying within the spirit and scope of the claims amended hereto.

Further aspects of the invention are defined in the following clauses:

1. A method of providing an electronic signature to a server, the method comprising:

receiving data for said server at a proxy system for said server;

reading said received data to identify a signature request;

obtaining a signature for a portion of data associated with said request, responsive to said request, and

providing said signature from said proxy system to said server.

2. A method as defined in clause 1 wherein said signature comprises a signature of a user of said server, the method further comprising:

authenticating said user prior to providing said signature.

3. A method as defined in clause 1 or 2 wherein said portion of data for which said signature is obtained comprises a portion of said received data.

4. A method as defined in clause 2 wherein said portion of data for which said signature is obtained comprises data received from said user at said proxy system.

5. A method as defined in clause 2 wherein said portion of data for which said signature is obtained comprises data received from said server at said proxy system and forwarded by said proxy system to said user.

6. A method as defined in any one of clauses 2 to 5 wherein said data receiving comprises receiving data from a plurality of different users; wherein a said signature is common to a group of said users; and further comprising providing a user identifier to said server in association with a said common signature.

7. A method as defined in any one of clauses 2 to 5 further comprising sending a request to said user, said request including said portion of received data to be signed, for confirmation of signing; and receiving confirmation data from said user; and wherein said signature obtaining is responsive to said confirmation data.

8. A method as defined in any preceding clause wherein said signature request comprises a signature request tag, the method further comprising:

receiving said signature request tag from said server at said proxy system in association with a request for additional data;

forwarding said signature request tag and said request for additional data to a remote system; and

receiving data comprising said signature request tag and said additional data from said remote system at said proxy system, said portion of data for signature comprising a portion of said additional data.

9. A method as defined in any preceding clause further comprising authenticating said proxy system to a sender of said data portion.

10. A method as defined in any preceding clause wherein said providing of said signature to said server additionally or alternatively comprises providing said signature to a second server.

11. A method as defined in any one of clauses 2 to 10 further comprising filtering data received from said server for providing to the said user, to control provision of data to said user for establishing connections other than via said proxy system.

12. A method of providing an electronic signature to a server, the method comprising:
receiving data to be signed from said server at a proxy system;
sending said data to be signed from said proxy system to a user;
receiving a signature request for said data to be signed at said proxy system;
obtaining a signature for said data to be signed responsive to said signature request, and
providing said signature from said proxy system to said server.

13. Computer program code to, when running, implement the method of any one of clauses 1 to 12.

14. A carrier carrying the program code of clause 13.

15. A computer system including a proxy system comprising the carrier of clause 13 and at least one said server, said proxy system and said at least one server being coupled by a secure link.

16. A data carrier carrying a computer network signal, the signal comprising a signature request tag in association with data for signature.

17. A method of authenticating electronic data supplied by a user of a source device to a remote device, the method comprising:

receiving input data including data for authentication from the source device;

identifying authentication request information in the input data;

obtaining authentication data for said data for authentication responsive to said identifying, and

outputting the authentication data for said remote device;

wherein the authentication data authenticates that the supplier of the data is the user.

18. A method as defined in clause 17 wherein said obtaining of authentication data comprises:

sending data to be authenticated to authentification apparatus, said data to be authenticated being determined from said input data; and

receiving said authentication data from said authenticating apparatus.

19. A method according to clause 13, wherein the authenticating apparatus comprises a signing apparatus and the authentication data comprises a digital signature.

20. A method according to any of clauses 17 to 19, the method further comprising:

outputting the input data to a said remote device.

21. A method according to any clauses 17 to 20, the method further comprising:

outputting the input data to a second remote device.

22. A computer program to, when running, carry out the method steps of any of clauses 17 to 21.

23. A computer readable medium storing the computer program of clause 22.

24. A data processing apparatus for processing electronic data supplied by a user of a source device to be signed, the apparatus comprising;

a communication interface for inputting and outputting data;

a data memory operable to store data to be processed;

an instruction memory storing processor implementable instructions; and

a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and

wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:

receive input data including data for authentication from a source device;

identifying authentication request information in the input data;

obtain authentication data for said data for authentication; and

output the authentication data for a recipient device;

wherein the authentication data authenticates that the supplier of the data is the user.

25. An apparatus according to clause 24, including certifying apparatus comprising a signing device, and wherein the certification data comprises a digital signature.

26. An apparatus according to clause 24 or 25, wherein the instructions stored in the instruction memory further comprise instructions for controlling the processor to output the input data to the recipient device.

27. An apparatus according to any of clauses 24 to 26, wherein the instructions stored in the instruction memory further comprise instructions for controlling the processor to output the input data to a second recipient device.

28. A method of supplying electronic data to a remote device for signing, the method comprising:

receiving data at a user terminal from a first remote device, said received data incorporating a request for user input data and signature request data;

inputting user data in accordance with said request; and

outputting to a second remote device data incorporating said user input data and said signature request data.

29. A method according to clause 28, wherein the first and second remote devices comprise the same device.

30. A computer program to, when running, carry out the method steps of clause 28 or 29.

31. A computer readable medium storing the computer program of clause 30.

32. A processing apparatus for processing electronic data to be signed, the apparatus comprising:

a communication interface for inputting and outputting data;

a data memory operable to store data to be processed;

an instruction memory storing processor implementable instructions; and

a processor operable to read and process the stored data in accordance with the instructions in the instruction memory; and

wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:

receive from a first remote device data incorporating a request for user input data and signature request data;

input user data in accordance with said request; and

output to a second remote device data incorporating said user input data and said signature request data.

33. An apparatus according to clause 32, wherein the first and second remote devices comprise the same device.

34. A reverse proxy device, the device comprising

means to authenticate an application to a user of a user terminal;

means to obtain an authentication for said user of said user terminal;

means to obtain a signature for user data; and

means to receive from said terminal user data for signature and a signature request, and responsive to said signature request, to obtain a signature for said user data and forward said signature and said user data to said application.

35. A proxy device as defined in clause 34 wherein said means to obtain an authentication for said user of said user terminal comprises means to send data to and receive data from an authentication server.

36. A proxy device as defined in clause 34 or 35 wherein said means to obtain a signature for user data comprises means to send data to and receive data from a signature server.

37. A proxy device as defined in clause 34, 35 or 36 wherein said means to authenticate said application comprises means to authenticate said proxy device to said user.

38. A proxy device as defined in clause 34, 35, 36 or 37 further comprising means to send a confirmation of said user data to be signed to said user, and to receive a user confirmation; and wherein said signing is responsive to said user confirmation.

39. A proxy device as defined in clause 38 wherein said user confirmation comprises authentication data for authentication of said user.

40. A proxy device as defined in any one of clauses 34 to 39 wherein a common key is employed for signing data of a plurality of users.

41. A proxy device as defined in any one of clauses 34 to 40 wherein the device is configured to operate in a public key infrastructure.

42. A proxy device as defined in any one of clauses 34 to 41 wherein the device comprises an Internet proxy server.

43. A proxy device as defined in clause 34 including a user authenticator and/or a user data signer, and wherein said means to obtain an authentication for said user of said user terminal and/or said means to obtain a signature for user data is configured to communicate with said user authenticator and/or user data signer.

44. A proxy device as defined in any one of clauses 34 to 43 wherein said means comprise computer program code.

45. A carrier carrying the computer program code of clause 44.

46. A signal, in particular for use with the reverse proxy device of any one of clauses 34 to 44, the signal including data for submission to a computer program requiring signed data, characterised in that the signal further includes a request for the data to be signed.

47. A carrier carrying the signal of clause 46.

48. A method of protecting a link between a user terminal and an application server via a proxy system for said application server, the method comprising:

receiving data from said application server at said proxy system;

determining whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and

forwarding said received data to said user terminal responsive to said determining.

49. A method as defined in clause 48 wherein said forwarding responsive to said determining comprises inhibiting said forwarding when said determining establishes that said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden.

50. A method as defined in clause 48 wherein said forwarding responsive to said determining comprises modifying said received data to remove said data for establishing a connection between said user terminal and an address to which connection is forbidden prior to said forwarding.

51. A method as defined in clause 48, 49 or 50 wherein said data for establishing a connection between said user terminal and an address to which connection is forbidden comprises data for establishing a connection with said user terminal other than via said proxy system.

52. A method as defined in any one of clauses 48 to 51 wherein said determining comprises comparing data in said received data with a list of permitted addresses.

53. A method as defined in any one of clauses 48 to 51 wherein said determining comprises comparing data in said received data with a list of forbidden addresses.

54. A method as defined in any one clauses 48 to 53 wherein said link comprises a packet data link and wherein said received data comprises packet payload data.

55. A method as defined in any one of clauses 48 to 54 wherein said determining comprises parsing said received data for a connection address.

56. A method as defined in any one of clauses 48 to 55, the method further comprising:

outputting audit data to an audit log responsive to said determining.

57. A method as defined in any one of clauses 48 to 56, the method further comprising:

outputting system management data responsive to said determining, wherein the system management data provides an alert when said determining establishes that said received data

includes data for establishing a connection between said user terminal and an address to which connection is forbidden.

58. A carrier carrying processor control code to, when running, implement the method of any one of clauses 48 to 57.

59. A reverse proxy server configured to implement the method of any one of clauses 48 to 57.

60. A method of providing a secure link between a user terminal and an application server, the method comprising:

establishing a secure link between said user terminal and a proxy system for said application server; and

using the method of any one of clauses 48 to 57 to protect said secure link.

61. Data processing apparatus for protecting a link between a user terminal and an application server via a proxy system for said application server, the apparatus comprising:

a communication interface for inputting and outputting data;

a data memory operable to store data to processed;

an instruction memory storing processor implementable instructions; and

a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and

wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:

receive data from said application server via said proxy system;

determine whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and

forward said received data via said proxy system to said user terminal responsive to said determining.

62. Data processing apparatus as defined in clause 61 configured to operate as a reverse proxy server.

CLAIMS:

1. A method of providing an electronic signature to a server, the method comprising:

receiving data for said server at a proxy system for said server;

reading said received data to identify a signature request;

obtaining a signature for a portion of data associated with said request, responsive to said request, and

providing said signature from said proxy system to said server.

2. A method as claimed in claim 1 wherein said signature comprises a signature of a user of said server, the method further comprising:

authenticating said user prior to providing said signature.

3. A method as claimed in claim 1 or 2 wherein said portion of data for which said signature is obtained comprises a portion of said received data.

4. A method as claimed in claim 2 wherein said portion of data for which said signature is obtained comprises data received from said user at said proxy system.

5. A method as claimed in claim 2 wherein said portion of data for which said signature is obtained comprises data received from said server at said proxy system and forwarded by said proxy system to said user.

6. A method as claimed in any one of claims 2 to 5 wherein said data receiving comprises receiving data from a plurality of different users; wherein a said signature is common to a group of said users; and further comprising providing a user identifier to said server in association with a said common signature.

7. A method as claimed in any one of claims 2 to 5 further comprising sending a request to said user, said request including said portion of received data to be signed, for

confirmation of signing; and receiving confirmation data from said user; and wherein said signature obtaining is responsive to said confirmation data.

8. A method as claimed in any preceding claim wherein said signature request comprises a signature request tag, the method further comprising:

receiving said signature request tag from said server at said proxy system in association with a request for additional data;

forwarding said signature request tag and said request for additional data to a remote system; and

receiving data comprising said signature request tag and said additional data from said remote system at said proxy system, said portion of data for signature comprising a portion of said additional data.

9. A method as claimed in any preceding claim further comprising authenticating said proxy system to a sender of said data portion.

10. A method as claimed in any preceding claim wherein said providing of said signature to said server additionally or alternatively comprises providing said signature to a second server.

11. A method as claimed in any one of claims 2 to 10 further comprising filtering data received from said server for providing to the said user, to control provision of data to said user for establishing connections other than via said proxy system.

12. A method of providing an electronic signature to a server, the method comprising:

receiving data to be signed from said server at a proxy system;

sending said data to be signed from said proxy system to a user;

receiving a signature request for said data to be signed at said proxy system;

obtaining a signature for said data to be signed responsive to said signature request, and

providing said signature from said proxy system to said server.

13. Computer program code to, when running, implement the method of any one of claims 1 to 12.

14. A carrier carrying the program code of claim 13.

15. A computer system including a proxy system comprising the carrier of claim 13 and at least one said server, said proxy system and said at least one server being coupled by a secure link.

16. A data carrier carrying a computer network signal, the signal comprising a signature request tag in association with data for signature.

17. A method of authenticating electronic data supplied by a user of a source device to a remote device, the method comprising:

receiving input data including data for authentication from the source device;

identifying authentication request information in the input data;

obtaining authentication data for said data for authentication responsive to said identifying, and

outputting the authentication data for said remote device;

wherein the authentication data authenticates that the supplier of the data is the user.

18. A method as claimed in claim 17 wherein said obtaining of authentication data comprises:

sending data to be authenticated to authentification apparatus, said data to be authenticated being determined from said input data; and

receiving said authentication data from said authenticating apparatus.

19. A method according to claim 13, wherein the authenticating apparatus comprises a signing apparatus and the authentication data comprises a digital signature.

20. A method according to any of claims 17 to 19, the method further comprising:

outputting the input data to a said remote device.

21. A method according to any claims 17 to 20, the method further comprising:
outputting the input data to a second remote device.

22. A computer program to, when running, carry out the method steps of any of claims 17 to 21.

23. A computer readable medium storing the computer program of claim 22.

24. A data processing apparatus for processing electronic data supplied by a user of a source device to be signed, the apparatus comprising;
a communication interface for inputting and outputting data;
a data memory operable to store data to be processed;
an instruction memory storing processor implementable instructions; and
a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and
wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:
receive input data including data for authentication from a source device;
identifying authentication request information in the input data;
obtain authentication data for said data for authentication; and
output the authentication data for a recipient device;
wherein the authentication data authenticates that the supplier of the data is the user.

25. An apparatus according to claim 24, including certifying apparatus comprising a signing device, and wherein the certification data comprises a digital signature.

26. An apparatus according to claim 24 or 25, wherein the instructions stored in the instruction memory further comprise instructions for controlling the processor to output the input data to the recipient device.

27. An apparatus according to any of claims 24 to 26, wherein the instructions stored in the instruction memory further comprise instructions for controlling the processor to output the input data to a second recipient device.

28. A method of supplying electronic data to a remote device for signing, the method comprising:

receiving data at a user terminal from a first remote device, said received data incorporating a request for user input data and signature request data;

inputting user data in accordance with said request; and

outputting to a second remote device data incorporating said user input data and said signature request data.

29. A method according to claim 28, wherein the first and second remote devices comprise the same device.

30. A computer program to, when running, carry out the method steps of claim 28 or 29.

31. A computer readable medium storing the computer program of claim 30.

32. A processing apparatus for processing electronic data to be signed, the apparatus comprising:

a communication interface for inputting and outputting data;

a data memory operable to store data to be processed;

an instruction memory storing processor implementable instructions; and

a processor operable to read and process the stored data in accordance with the instructions in the instruction memory; and

wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:

receive from a first remote device data incorporating a request for user input data and signature request data;

input user data in accordance with said request; and

output to a second remote device data incorporating said user input data and said signature request data.

33. An apparatus according to claim 32, wherein the first and second remote devices comprise the same device.

34. A reverse proxy device, the device comprising

means to authenticate an application to a user of a user terminal;

means to obtain an authentication for said user of said user terminal;

means to obtain a signature for user data; and

means to receive from said terminal user data for signature and a signature request, and responsive to said signature request, to obtain a signature for said user data and forward said signature and said user data to said application.

35. A proxy device as claimed in claim 34 wherein said means to obtain an authentication for said user of said user terminal comprises means to send data to and receive data from an authentication server.

36. A proxy device as claimed in claim 34 or 35 wherein said means to obtain a signature for user data comprises means to send data to and receive data from a signature server.

37. A proxy device as claimed in claim 34, 35 or 36 wherein said means to authenticate said application comprises means to authenticate said proxy device to said user.

38. A proxy device as claimed in claim 34, 35, 36 or 37 further comprising means to send a confirmation of said user data to be signed to said user, and to receive a user confirmation; and wherein said signing is responsive to said user confirmation.

39. A proxy device as claimed in claim 38 wherein said user confirmation comprises authentication data for authentication of said user.

40. A proxy device as claimed in any one of claims 34 to 39 wherein a common key is employed for signing data of a plurality of users.

41. A proxy device as claimed in any one of claims 34 to 40 wherein the device is configured to operate in a public key infrastructure.

42. A proxy device as claimed in any one of claims 34 to 41 wherein the device comprises an Internet proxy server.

43. A proxy device as claimed in claim 34 including a user authenticator and/or a user data signer, and wherein said means to obtain an authentication for said user of said user terminal and/or said means to obtain a signature for user data is configured to communicate with said user authenticator and/or user data signer.

44. A proxy device as claimed in any one of claims 34 to 43 wherein said means comprise computer program code.

45. A carrier carrying the computer program code of claim 44.

46. A signal, in particular for use with the reverse proxy device of any one of claims 34 to 44, the signal including data for submission to a computer program requiring signed data, characterised in that the signal further includes a request for the data to be signed.

47. A carrier carrying the signal of claim 46.

48. A method of protecting a link between a user terminal and an application server via a proxy system for said application server, the method comprising:

receiving data from said application server at said proxy system;

determining whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and

forwarding said received data to said user terminal responsive to said determining.

49. A method as claimed in claim 48 wherein said forwarding responsive to said determining comprises inhibiting said forwarding when said determining establishes that said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden.

50. A method as claimed in claim 48 wherein said forwarding responsive to said determining comprises modifying said received data to remove said data for establishing a connection between said user terminal and an address to which connection is forbidden prior to said forwarding.

51. A method as claimed in claim 48, 49 or 50 wherein said data for establishing a connection between said user terminal and an address to which connection is forbidden comprises data for establishing a connection with said user terminal other than via said proxy system.

52. A method as claimed in any one of claims 48 to 51 wherein said determining comprises comparing data in said received data with a list of permitted addresses.

53. A method as claimed in any one of claims 48 to 51 wherein said determining comprises comparing data in said received data with a list of forbidden addresses.

54. A method as claimed in any one claims 48 to 53 wherein said link comprises a packet data link and wherein said received data comprises packet payload data.

55. A method as claimed in any one of claims 48 to 54 wherein said determining comprises parsing said received data for a connection address.

56. A method as claimed in any one of claims 48 to 55, the method further comprising:

outputting audit data to an audit log responsive to said determining.

57. A method as claimed in any one of claims 48 to 56, the method further comprising:

outputting system management data responsive to said determining, wherein the system management data provides an alert when said determining establishes that said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden.

58. A carrier carrying processor control code to, when running, implement the method of any one of claims 48 to 57.

59. A reverse proxy server configured to implement the method of any one of claims 48 to 57.

60. A method of providing a secure link between a user terminal and an application server, the method comprising:

establishing a secure link between said user terminal and a proxy system for said application server; and

using the method of any one of claims 48 to 57 to protect said secure link.

61. Data processing apparatus for protecting a link between a user terminal and an application server via a proxy system for said application server, the apparatus comprising:

a communication interface for inputting and outputting data;

a data memory operable to store data to processed;

an instruction memory storing processor implementable instructions; and

a processor coupled to said data memory and to said instruction memory and operable to read and process the stored data in accordance with the instructions in the instruction memory; and

wherein the instructions stored in the instruction memory comprise instructions for controlling the processor to:

receive data from said application server via said proxy system;

determine whether said received data includes data for establishing a connection between said user terminal and an address to which connection is forbidden; and

forward said received data via said proxy system to said user terminal responsive to said determining.

62. Data processing apparatus as claimed in claim 61 configured to operate as a reverse proxy server.