(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
27 September 2007 (27.09.2007)

PCT

(10) International Publication Number
WO 2007/106967 A1

(51) International Patent Classification:
H04L 29/06 (2006.01) *H04L 9/00* (2006.01)
H04L 12/16 (2006.01)

(21) International Application Number:
PCT/CA2006/000434

(22) International Filing Date: 23 March 2006 (23.03.2006)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **SLIPSTREAM DATA INC.** [CA/CA]; 12-50 Bathurst Drive, Waterloo, Ontario N2V 2C5 (CA).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **NANDURI, Akshay** [CA/CA]; 502-55 William Street East, Waterloo, Ontario N2J 4Z1 (CA). **SINGH, Ajit** [CA/CA]; 577 Sandbury Lane, Mississauga, Ontario N2T 1Z5 (CA). **AHMED, Salmaan** [CA/CA]; 949 Blizzard Road, Mississauga, Ontario L5V 1T1 (CA). **SZE, David** [CA/CA]; 712-205 Victoria Street South, Kitchener, Ontario N2G 4Z6 (CA).

(74) Agent: **BERESKIN & PARR**; 40 King Street West, Toronto, Ontario M5H 3Y2 (CA).

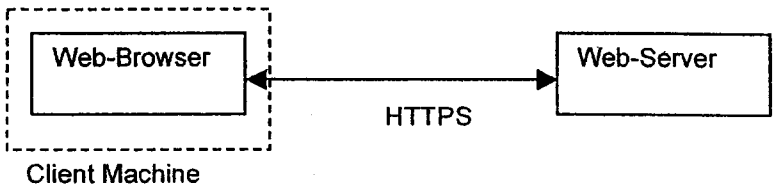
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** A BROWSER-PLUQIN BASED METHOD FOR ADVANCED HTTPS DATA PROCESSING



(57) **Abstract:** The invention described here deals with implementing custom data processing of HTTPS based on a Browser-Plugin Method. Such custom data processing may include, but is not limited to, custom data compression, custom data encryption, data monitoring, data modification. There are two distinct methods to implement the Browser-Plugin Method for Advanced HTTPS Data Processing of the subject invention (BPAHDP). In both cases, BPAHDP provides the option of conducting custom data processing that co-exists with data compression, data encryption, or other types of data processing operations supported by the HTTP standard. Additionally, both BPAHDP methods ensure that the web-browser still implements and executes the underlying SSL/TLS channel setup and encryption operations. In both embodiments of BPAHDP, the most critical functionality is the ability to modify HTTP request/response headers and data sent over a TLS/SSL channel. In the regular HTTP case (HTTP over TCP) headers and data are sent as clear-text (i.e., as unencrypted data). Therefore, any HTTP proxy component can intercept and modify header/data as it chooses - allowing custom data processing operations (including a custom compression operation) to be implemented. For HTTPS traffic, the data leaving a web-browser is encrypted. Therefore, a proxy cannot modify encrypted data, hence the novelty of the BPAHDP methodology. Both methods require specific implementation methods that are described. In particular, both embodiments of BPAHDP require specific techniques to facilitate the use of Microsoft Internet Explorer as a BPAHDP enabled web-browser. Microsoft COM (Component Object Model) interfaces and IE's Pluggable Protocol capabilities are utilized to meet all requirements of both BPAHDP embodiments.

WO 2007/106967 A1

Title: A Browser-Plugin Based Method For Advanced HTTPS Data Processing

Field of the invention

[0001] The present invention relates to a novel method of utilizing a browser plugin that provides a technique for interception and further processing of data sent via the HTTPS protocol. The HTTPS protocol is defined as HTTP over a Secure Socket Layer (SSL), or HTTP over a Transport Layer Security (TLS). See A. Freier, P. Karlton and P. Kocher, "Internet-Draft: The SSL Protocol Version 3.0," *Transport Layer Security Group*, November 1996, for a discussion of SSL, and T. Dierks and C. Allen, "Request for Comments: 2246 – The TLS Protocol," *Network Working Group*, January 1999 for a discussion of TLS. A potential application of the subject method can be to apply proprietary data compression methods to reduce the volume of data communication of HTTPS payload data and also to possibly reduce the data transmission time. It should be noted that the option of applying proprietary data processing in this case is available in addition to standard built-in HTTPS compression and encoding approaches such as 'Content-Encoding' methods: gzip, compress, deflate, described in section 3.5 of the article entitled "Request for Comments: 2616, Hypertext Transfer Protocol – HTTP/1.1," *Network Working Group*, June 1999, by R. Fielding, J. Gettys, J. Mogul, H Frystyk, L. Masinter, P. Leach, and T. Berners-Lee.

Background of the invention

[0002] Presently, large volumes of data are delivered over the Internet by server computers to client computing devices such as desktop computers, laptops and various handheld digital devices using a communication protocol called 'Hyper Text Transfer Protocol' (HTTP). The HTTP protocol strictly defines the interaction between a client device that sends requests for data, and a server that supplies the data. A client, after sending the request for data to the server, waits for the server's response, and then normally, upon receipt of data, delivers

the data to the end user. In many cases, the client is implemented by a software component called a 'web-browser'. The server is usually implemented by a software component called a 'web-server'.

[0003] Rapid expansion in Internet usage by businesses, banking and direct consumer shopping led to the definition of a standard approach for sending encrypted HTTP data between HTTP clients and servers. This approach (also known as HTTPS) was first implemented by Netscape as HTTP over a Secure-Socket Layer (SSL) TCP/IP connection. The HTTPS protocol allowed end-users and corporations to safely send credit-card and other sensitive information over the internet. More specifically, it prevents eavesdropping, message forgery and tampering of HTTP data sent between client/server applications. The first implementations of HTTPS utilized 40-bit encryption while the latest standard (HTTP over TLS described in an article by E. Rescorla entitled "Request for Comments: 2818, HTTP over TLS," *Network Working Group*, May 2000), facilitates the use of powerful 128-bit encryption.

[0004] The underlying implementations of SSL and TLS are described in the article by A. Freier et al. and in the article by T. Dierks et al. respectively. Although the mechanisms are different, both SSL and TLS essentially involve several common stages:

1) Protocol version identification between web-browser and web-server.
2) Transmission of web-server's public key to the web-browser (the signed public key is also known as a *Certificate*).
3) Web-browser verifies the *Certificate* by communicating with a trusted entity (known as a Root Certificate Authority). This ensures that the web-browser is communicating with the intended web-server.
4) Negotiation and exchange of 'symmetric encryption-key' between web-browser and web-server.
5) The symmetric encryption-key is then used to establish an SSL or TLS 'channel' between the web-browser and web-server.

6) Encryption of all HTTP payload data between web-browser and web-server over the SSL/TLS channel.

[0005] It is imperative for SSL and TLS functionality to be implemented by the web-browser so that the SSL/TLS 'channel' is established directly from the web-browser. This guarantees the encryption and security of all HTTP data originating from the web-browser and received from the web-server. Reference is made to Figure 1 which depicts the aforementioned scenario.

[0006] Now consider the compression of HTTPS data (HTTP payload data sent over an SSL or TLS channel) in order to reduce the volume of data transmission. Content-encoding methods inherently present in the HTTP standard (section 3.5 of the article by R. Fielding et al.) can be used to reduce the volume of data sent in HTTPS transactions. These methods consist of a class of lossless compression algorithms such as gzip, compress and deflate which can be supported within web-browsers and web-servers.

[0007] Unfortunately, custom approaches (e.g., proprietary data-compression) or any other proprietary or custom data processing cannot be supported by HTTPS. HTTPS does not provide for a standard interface or mechanism to facilitate custom data transformation. More specifically, the encryption of HTTP data actually randomizes the original source HTTP data. In an information-theoretic sense, the entropy of the encrypted data is significantly higher than the original source data. Significant randomization of source data limits the effectiveness of data-compression. The encrypted data also makes it difficult to do many other types of desirable data processing operations such as data recording, data monitoring, or data alteration. Since SSL and TLS are designed to prevent data-tampering or "man in the middle" viewing, retrieving the original source HTTP data is extremely difficult.

Summary of the invention

[0008] There are two distinct methods to implement the Browser-Plugin Method for Advanced HTTPS Data Processing of the subject invention

(BPAHDP). In both cases, BPAHDP provides the option of conducting custom data processing operations including data compression, data encryption, or other types of data processing operations supported by the HTTP standard. Additionally, both BPAHDP methods ensure that the web-browser still implements and executes the underlying SSL/TLS channel setup and encryption operations. In both embodiments of BPAHDP, the most critical functionality is the ability to modify HTTP request/response headers and data sent over a TLS/SSL channel. In the regular HTTP case (HTTP over TCP) headers and data are sent as clear-text (i.e., as unencrypted data). Therefore, an HTTP proxy (an intermediary software entity that fetches HTTP data on behalf of a group of HTTP clients) component can intercept and modify header/data as it chooses – allowing custom data processing operations (including a custom compression operation) to be implemented. The subject of Performance Enhancing Proxies is discussed in a document (Request for Comments 3135, Network Working Group, June 2001) by J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, entitled, "Performance Enhancing Proxies Intended to Mitigate Link-Related (Degradations". For HTTPS traffic, the data leaving a web-browser is encrypted. Therefore, a proxy cannot modify encrypted data, hence the novelty of the BPAHDP methodology. Both BPAHDP methods require specific implementation methods. In particular, both embodiments of BPAHDP require specific techniques to facilitate the use of Microsoft Internet Explorer as a BPAHDP enabled web-browser. Microsoft COM (Component Object Model) interfaces and IE's Pluggable Protocol capabilities are utilized to meet all requirements of both BPAHDP embodiments.

Brief description of the drawings

[0009] Figure 1 is a diagram depicting a SSL/TLS channel between a web-browser and a web-server.

[0010] Figure 2 is a diagram depicting the block level architecture for implementing Method '1' described in this document.

[0011] Figure 3 is a diagram depicting the block level architecture for implementing Method '2' described in this document.

Detailed description of the invention

Embodiment of BPAHDP Method '1'

[0012] Define WSA as a BPAHDP enabled web-server (a web-server that has implemented 'Advanced Processing' or custom data processing of HTTP payload data) that is able to accept HTTPS connections from standard web-browsers. In addition, define WBA as a web-browser that has implemented the following functionality required by BPAHDP, and is thus BPAHDP enabled:

[0013] 1) The ability to add custom headers on all outgoing HTTP requests sent over an SSL/TLS channel. For example:

```
GET http://www.slipstream.com HTTP/1.0\r\n
Connection: Close\r\n
\r\n
```

is modified to:

```
GET http://www.slipstream.com HTTP/1.0\r\n
Connection: Close\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

[0014] 2) The presence of the X-BPAHDP header identifies WBA as being BPAHDP enabled. The 'control_info' field is utilized to identify the BPAHDP version (identifying the supported data processing operations present in the web-browser) and any other relevant control information required during the custom data processing operation.

[0015] 3) The ability to read and modify the HTTP response header returned by WSA over the TLS/SSL channel. For example:

```
HTTP/1.0 200 OK\r\n
```

```
Content-Type: text/xml\r\n
Content-Length: 300\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

might be modified to:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Content-Length: 200\r\n
\r\n
```

[0016] Since BPAHDP facilitates data transformation and filtering, the modification of certain HTTP headers (Content-Type and Content-Length) may be required. Additionally, certain response headers may need to be parsed and stored by the BPAHDP filtering method (in the above example <control_info> may be used during the decompression or some ther data processing operation).

[0017] 4) The ability to read and modify the HTTP payload data returned by WSA over an SSL/TLS channel. For example:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain\r\n
Content-Length: 16\r\n
X-BPAHDP: <control_info>\r\n
\r\n
"This is a test"
```

may be modified to

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain \r\n
Content-Length: 20\r\n
\r\n
"This is not a test"
```

[0018] 5) The BPAHDP enabled web-browser must be able to communicate with other non-BPAHDP enabled web-servers for both HTTP and HTTPS data.

[0019] System Architecture and Operation of the Method '1'

This subsection describes the block level architecture and operational method for carrying out custom data processing on HTTPS data based on the capabilities described earlier in Method '1'. The terms WBA and WSA in this subsection have the same definition as given in Method '1'. In the method described in this subsection, the HTTP based data exchange between WBA and WSA takes place over a SSL/TLS channel. Block level architectural diagram depicting the operation is given in Figure 2.

1) The BPAHDP enabled web-browser (WBA) is used to modify the outgoing HTTP request headers to indicate to the BPAHDP enabled web-server (WSA) that the requesting WBA is capable of carrying out certain custom data processing (e.g. ability to decompress data that is compressed using a certain data compression algorithm called 'X')

2) If the WSA receives the indication from a WBA about being BPAHDP enabled and its associated capabilities, the WSA sends the HTTP response data to WBA in a form that can only be processed by a WBA that possesses the indicated capabilities. For example, if the WBA indicated that it is capable of decompressing data that has been compressed using a certain data compression algorithm called 'X', then WSA compresses its HTTP payload data using the algorithm 'X'. If the WSA does not receive any such indication or if the WSA is not BPAHDP enabled then WSA sends its HTTP response in the usual manner intended for a normal web-browser and without using the algorithm 'X'.

3) The ability of WBA to modify HTTP response headers and payload is used by the WBA to intercept the HTTP response headers and data and apply the custom data processing operation and present it to the web-browser in a format that is acceptable to a normal web-browser.

Embodiment of BPAHDP Method '2'

[0020] Define WSA as a BPAHDP enabled web-server (a web-server that has implemented 'Advanced Data Processing' or custom data processing of HTTP payload data) and is able to accept HTTPS connections from standard web-browsers. Also define CS as a standard HTTP/HTTPS web-server that provides the HTTP content to WSA.

[0021] The BPAHDP enabled web-server (WSA) meets the following requirements:

1) Ability to communicate with the CS via HTTP or HTTPS.

2) Executes the compression or filtering operation on HTTP payload data.

3) Can receive and accept HTTPS requests from WBA or from non-BPAHDP enabled web-browsers.

[0022] In addition, define WBA as a web-browser that has implemented the following functionality required by BPAHDP:

[0023] 1) The ability to add custom headers on all outgoing HTTP requests sent over an SSL/TLS channel. For example:

```
GET http://www.slipstream.com HTTP/1.0\r\n
Connection: Close\r\n
\r\n
```

is modified to:

```
GET http://www.slipstream.com HTTP/1.0\r\n
Connection: Close\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

[0024] 2) The presence of the X-BPAHDP header identifies WBA as being BPAHDP enabled. The <control_info> field is utilized to identify the BPAHDP version (identifying the supported custom data processing operations present in the web-browser) and any other relevant control information used during the data processing operations.

[0025] 3) The ability to read and modify the HTTP response header returned by WSA over the TLS/SSL channel. For example:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/xml\r\n
Content-Length: 300\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

may be modified to:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Content-Length: 200\r\n
\r\n
```

[0026] Since BPAHDP facilitates data transformation and filtering, the modification of certain HTTP headers (Content-Type and Content-Length) may be required. Additionally, certain response headers may need to be parsed and stored by the BPADPH filtering method (in the above example <control_info> may be used during the decompression or filtering operation).

[0027] 4) The ability to read and modify the HTTP payload data returned by WSA over an SSL/TLS channel. For example:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain\r\n
Content-Length: 16\r\n
X-BPACH: <control_info>\r\n
\r\n
"This is a test"
```

may be modified to

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain \r\n
Content-Length: 20\r\n
\r\n
"This is not a test"
```

[0028] 5) The ability to modify the HTTP URL (Uniform Resource Locator) of objects originally destined for the CS. For example, in order to route all HTTP requests to WSA :

```
GET http://www.slipstreamCS.com HTTP/1.0\r\n
Connection: Close\r\n
\r\n
```

is modified to:

```
GET http://www.slipstreamWSA.com HTTP/1.0\r\n
Connection: Close\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

[0029] 6) The BPAHDP enabled web-browser must be able to communicate with other non-BPAHDP enabled web-servers for both HTTP and HTTPS data.

System Architecture and Operation of the Method '2':

[0030] In certain situations, it may not be possible to enable the target content server (CS) with the custom data processing capability required for

BPAHDP. In that case, the HTTPS request originally intended for CS can be redirected to another server (WSA) that is enabled with custom data processing expected by a BPAHDP-enabled web-browser (WBA). This subsection describes the system architecture and operational method for carrying out custom data processing on HTTPS data based on the capabilities described earlier in Method '2'. The terms CS, WBA and WSA in this subsection have the same definition as given in Method '2'. In the method described in this subsection, the HTTP based data exchange between WBA and WSA takes place over a SSL/TLS channel. The use of SSL/TLS channel for the HTTP based data exchange between WSA and CS is optional. System architecture depicting the operation of the method is given in Figure 3.

1) The ability to modify the outgoing URL in Method '2' is used by WBA to divert a HTTP request over SSL/TLS channel originally directed to a content-server CS to another web-server WSA that is enabled with the data processing capabilities expected by a BPAHDP enabled web browser in Method '2'.

2) The BPAHDP enabled web-browser modifies the outgoing HTTP request headers over SSL/TLS channel to indicate to the WSA that the requesting WBA is capable of carrying out certain custom data processing (e.g. ability to decompress data that is compressed using an algorithm called X) WBA also specifies the URL of the requested data at content server CS.

3) WSA receives the request from the WBA over SSL/TLS channel and makes a request to the content server CS for the response data required by WBA using HTTP or HTTPS and receives the response data from the content server CS.

4) Based on the indication from the requesting WBA about its capabilities, the WSA sends the HTTP response data over SSL/TLS channel to the

WBA in a form that can only be processed by a WBA that possesses the indicated capabilities. For example, if the WBA indicated that it is capable of decompressing data that has been compressed using a data compression algorithm called 'X', then WSA compresses is HTTP payload data to be sent over SSL/TLS channel using the algorithm 'X'. If the WSA does not receive any such indication then WSA sends its HTTP response over SSL/TLS channel for a normal web-browser that may not be BPAHDP enabled.

5) The ability of WBA to modify HTTP response headers and payload over SSL/TLS channel is used by the WBA to intercept the HTTP response headers and data and apply the custom data processing operation and present it to the web-browser in a format that is acceptable to a normal web-browser.

Implementation of Custom Data Processing Using BPAHDP Method

[0031] Implementation of BPAHDP capabilities on the server side to realize a WSA, required for Method 1 or Method 2, can be done in two ways: (1) Implementation of a custom web server that implements the specified capabilities, (2) Via implementation of server-side plug-in supported by several web-servers.

[0032] On the client side, the exact strategy used for implementing Method 1 and Method '2', is dependent on the Application Programmers' Interface (API) of the web browser product that is used for implementation. There are two distinct strategies for the implementation depending on the situation:

(a) This strategy is applicable in the case where the source code of the web-browser is available. In this case, intercepting outgoing HTTP requests, incoming HTTP response headers, HTTP payload data, is a straightforward programming task. The

interception and modification takes place after the browser has prepared the HTTPS GET request but before the request data has been encrypted. After interception, the modifications required by Method '1' or Method '2', as the case may be, can be applied.

(b) For some web browsers, access to the source code may not be available. In such cases, the available Application Programmers' Interfaces (APIs) are to be used. The following subsection describes the APIs for Microsoft's Internet Explorer versions 4.0 and up-to the versions released to-date that can be used for implementing Method '1' and Method '2':

Embodiment of BPAHDP Methods in Microsoft Internet Explorer

[0033] Define COM as Microsoft Component Object Model - Microsoft's implementation of interconnecting software objects via Remote Procedure Calls (RPC), function parameter marshalling and automation. Define IE as Microsoft Internet Explorer versions 4.0 and above. Define a Pluggable Protocol as a COM object that can override IE's handling of specific web schema. For example, an "http://" Pluggable Protocol can override a complete "http://" transaction that is normally carried out by IE.

[0034] The aforementioned requirements of BPAHDP can be met in IE by novel use of the COM interfaces exposed by Internet Explorer (IE). Use of these interfaces facilitate the modification of HTTP request headers, modification of HTTP response headers, modification of HTTP response data as well as URL link translation. Define the BPAHDP-PP as a COM object contained in a dynamic-linked library (DLL). The BPAHDP-PP is registered as an "https" Pluggable Protocol and implements all of core BPAHDP functionality (header modification, data processing, URL translation).

[0035] The following steps are followed to implement BPAHDP functionality in Microsoft Internet Explorer (IE):

1) Create a COM object that implements the IInternetProtocol, IInternetProtocolRoot and IHttpNegotiate interfaces. This COM object is the BPAHDP-PP.

2) Register the COM object as a Pluggable Protocol for the "https" protocol. This is accomplished by placing the unique identifier (GUID) of the BPAHDP-PP in the Microsoft Windows registry key - HKEY_CLASSES_ROOT\ PROTOCOLS\Name-Space Handler\https.

3) In the BPAHDP-PP, utilize the IInternetProtocolRoot::Start method to implement the URL link translation requirement of BPAHDP. For example, in Method-2 of BPAHDP, the URL requested by the IE web-browser may be modified from www.slipstreamCS.com to www.slipstreamWSA.com. The original URL to fetch is passed as a function parameter to IInternetProtocolRoot::Start. Hence, the actual URL fetched can be modified by the BPAHDP-PP, which provides the function implementation of IInternetProtocolRoot::Start.

4) In the BPAHDP-PP, utilize the IHttpNegotiate:: BeginningTransaction method to modify the outgoing HTTP request header sent over the SSL/TLS channel. The HTTP request header should be modified such that the IE web-browser is identified as being BPAHDP enabled. For example,

```
GET http://www.slipstreamWSA.com HTTP/1.0\r\n
Connection: Close\r\n
\r\n
```

is modified to:

```
GET http://www.slipstreamWSA.com HTTP/1.0\r\n
Connection: Close\r\n
X-BPAHDP: <control_info>\r\n
\r\n
```

The 'pszAdditionalHeaders' function parameter (a reference parameter) in IHttpNegotiate::BeginningTransaction is modified by the BPAHDP-PP to contain any custom HTTP headers to add in the outbound HTTP request.

5) In the BPAHDP-PP, utilize the IHttpNegotiate::OnResponse method to modify the incoming HTTP response header received over the SSL/TLS channel. The response header may be modified to reflect the specific data filtering operation that is implemented. The operation may change the MIME-type (content-type) of the data, as well as the aggregate content length. Moreover, the response header may contain specific control information for the filtering operation implemented in BPAHDP-PP. For example,

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain\r\n
Content-Length: 16\r\n
X-BPACH: <control_info>\r\n
\r\n
"This is a test"
```

may be modified to:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain \r\n
Content-Length: 20\r\n
\r\n
"This is not a test"
```

The 'szReponseHeaders' function parameter in IHttpNegotiate::OnResponse contains the entire HTTP response header returned by the BPAHDP enabled web-server. The 'szReponseHeaders' parameter can also be completely re-written by the BPAHDP-PP to contain a new HTTP response header.

6) The IInternetProtocol::Read method is called when the IE web-browser requests unfiltered (or decoded) data to be returned by the BPAHDP-PP. Data is written to a fixed-length buffer function parameter named 'pv', and the length of the unfiltered data is written passed back in the reference parameter 'pcbRead'.

[0036] It should be recognized that the embodiments described herein and shown in the drawing figures are meant to be illustrative only and should not be taken as limiting the scope of invention. Those skilled in the art will recognize that the elements of the illustrated embodiments can be modified in arrangement and detail without departing from the spirit of the invention. Therefore, the invention as described herein contemplates all such embodiments and modified embodiments as may come within the scope of the following claims or equivalents thereof.

CLAIMS:

1. A method for custom processing HTTPS data, comprising the steps of:
   a) creating a custom request header indicating that a web browser supports preselected customized processing operations;
   b) sending the custom request header with a HTTP request over a secure communication channel to a web server;
   c) receiving from the web server over the secure communication channel processed payload data and a HTTP response header correlatable therewith, wherein the processed payload data is created by processing original payload data based upon one or more of the customized processing operations supported by the web browser;
   d) modifying the response header to create a modified response header;
   e) modifying the processed payload data utilizing one or more of the customized processing operations to create modified payload data indicative of the original payload data; and
   f) presenting the modified header and the modified payload data to the web browser.

2. The method of claim 1, also comprising the step of modifying a URL of an object to be fetched by the web browser over the secure communication channel.

3. The method of claim 2, wherein the object is a web site address for a content server.

4. The method of claim 1, wherein the secure communication channel is a Secure Socket Layer (SSC) connection or Transport Layer Security (TLC) connection.

5. The method of claim 1, wherein the web browser is a standard web browser

6. The method of claim 1, wherein the customized processing operations comprise, or are selected from the group comprising: a compression operation, a decompression operation, an encryption operation, and a decryption operation.

7. A method for customized processing of HTTPS data, comprising the steps of:

a) sending a custom request header with a HTTP request over a secure communication channel to a web wherein the custom request header indicates that the web browser supports customized processing operations;

b) receiving from the web server over the secure communication channel processed payload data and a HTTP response header correlatable therewith, wherein the processed payload data is created by processing original payload data based upon one or more of the customized processing operations supported by the web browser; and

c) modifying the processed payload data utilizing one or more of the customized processing operations to create modified payload data indicative of the original payload data.

8. The method of claim 7, also comprising the step of modifying a URL of an object to be fetched by the web browser over the secure communication channel.

9. The method of claim 7, wherein the secure communication channel is a Secure Socket Layer (SSL) connection or a Transport Layer Security (TLS) connection.

10. The method of claim 7, wherein the customized processing operations comprise, or are selected from the group comprising: a compression operation, a decompression operation, an encryption operation, and a decryption operation.

11. A web browser enabled for customized processing of HTTPS data, comprising means for performing the steps of:

a) creating a custom request header indicating that the web browser supports preselected customized processing operations;

b) sending the custom request header with a HTTP request over a secure communication channel to a web server;

c) receiving from the web server over the secure communication channel processed payload data and a HTTP response header correlatable therewith, wherein the processed payload data is created by processing original payload data based upon one or more of the customized processing operations supported by the web browser; and

d) modifying the processed payload data utilizing one or more of the customized processing operations to create modified payload data indicative of the original payload data.

12. The method of claim 11, also comprising means for modifying a URL of an object to be fetched by the web browser over the secure communication channel.

13. The web browser of claim 11, wherein the secure communication channel is a Secure Socket Layer (SSL) connection or a Transport Layer Security (TLS) connection.

14. The web browser of claim 11, wherein customized processing operations comprise, or are selected from the group comprising: a compression operation, a decompression operation, an encryption operation, and a decryption operation.

15. A web browser plug-in for enabling a web browser to undertake the customized processing of HTTPS data, comprising means for modifying the web browser to perform the steps of:

a) modifying an outgoing request header to create a custom request header indicating that the web browser supports preselected customized processing operations;

b) sending the custom request header with a HTTP request over a secure communication channel to a web server;

c) receiving from the web server over the secure communication channel processed payload data and a HTTP response header correlatable therewith, wherein the processed payload data is created by processing original payload data utilizing one or more of the customized processing operations supported by the web browser;

d) modifying the response header to create a modified response header;

e) modifying the processed payload data utilizing one or more of the customized processing operations to create modified payload data indicative of the original payload data; and

f) presenting the modified header and the modified payload data to the web browser.

16. The web browser plug-in of claim 15, wherein the secure communication channel is a Secure Socket Layer (SSL) connection or a Transport Layer Security (TLS) connection.

17. The web browser plug-in of claim 15, wherein the web browser is Internet Explorer.

18. The web browser plug-in of claim 15, wherein the customized processing operations comprise, or are selected from the group comprising: a compression operation, a decompression operation, an encryption operation, and a decryption operation.

19. A computer program product for use on a computer system for customized processing of HTTPS data, the computer program product comprising:

a recording medium for recording means for instructing the computer system to perform the steps of:

a) creating a custom request header indicating that a web browser supports preselected customized processing operations;

b) sending the custom request header with a HTTP request over a secure communication channel to a web server;

c) receiving from the web server over the secure communication channel processed payload data and a HTTP response header correlatable therewith, wherein the processed payload data is created by processing original payload data based upon one or more of the customized processing operations supported by the web browser;

d) modifying the response header to create a modified response header;

e) modifying the processed payload data utilizing one or more of the customized processing operations to create modified payload data indicative of the original payload data;

f) presenting the modified header and the modified payload data to the web browser.

20. The computer program product of claim 19, wherein the computer program product is a web browser plug-in.

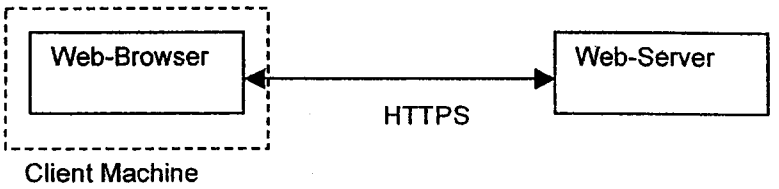


Figure 1

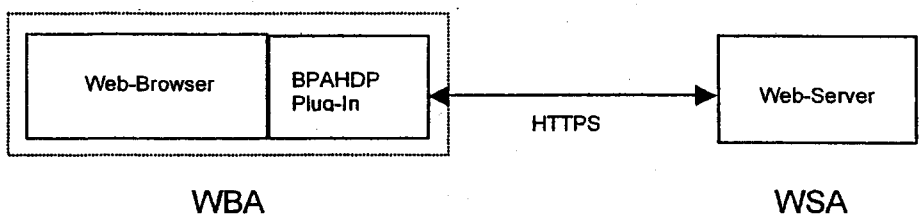


Figure 2

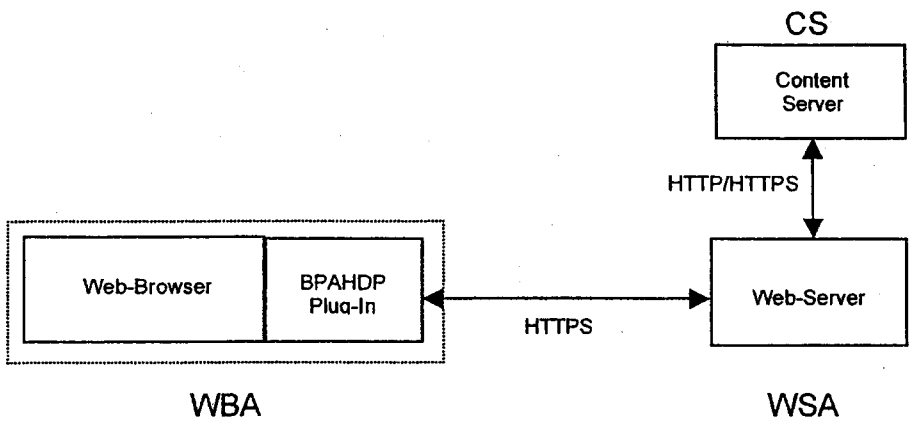

CS
Content Server
HTTP/HTTPS
Web-Browser
BPAHDP Plug-In
HTTPS
Web-Server
WBA
WSA

Figure 3

Date of receipt: 23 December 2006 (23.12.2006) PCT/CA2006/000434

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CA2006/000434

A. CLASSIFICATION OF SUBJECT MATTER

IPC: ***H04L 29/06*** (2006.01) , ***H04L 12/16*** (2006.01) , ***H04L 9/00*** (2006.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC: ***H04L 29/06*** (2006.01) , ***H04L 12/16*** (2006.01) , ***H04L 9/00*** (2006.01)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic database(s) consulted during the international search (name of database(s) and, where practicable, search terms used)

Delphion -- US Patents & Applications, European Patents & Applications, PCT Patents, European Abstracts

Search Terms -- data table, grouping, subset, database, retail

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6,658,006, Chen et al., H04L 12/56, 02-12-2003 *see entire document	1-20
Y	US 6,981,195, Newcombe et al., H03M 13/03, 27-12-2005 *see entire document	1-20
A	WO 2004/043042, Tsyganskiy, H04L 29/06, 21-05-2004 *see entire document	1-20
A	US 6,654,807, Farber et al., G06F 15/173, 25-11-2003 *see entire document	1-20
A	WO 2004/079497, Toutitou et al., G06F, 16-09-2004 *see entire document	1-20
A	US 2002/0002636, Vange et al., G06F 9/00, 03-01-2002 *see entire document	1-20

[ ] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
24 October 2006 (24-10-2006)	19 December 2006 (19-12-2006)
Name and mailing address of the ISA/CA Canadian Intellectual Property Office Place du Portage I, C114 - 1st Floor, Box PCT 50 Victoria Street Gatineau, Quebec K1A 0C9 Facsimile No.: 001(819)953-2476	Authorized officer Roger Collier 819- 956-9812

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CA2006/000434

Patent Document Cited in Search Report	Publication Date	Patent Family Member(s)	Publication Date
US6658006	02-12-2003	AU5176800 A	28-12-2000
		WO0076258 A1	14-12-2000
US6981195	27-12-2005	CN1685619 A	19-10-2005
		DE60308197D D1	19-10-2006
		EP1532741 A1	25-05-2005
		WO2004013973 A1	12-02-2004
WO2004043042	21-05-2004	AU2003279987 A1	07-06-2004
		EP1561327 A1	10-08-2005
		US2004088408 A1	06-05-2004
US6654807		AT245834T T	15-08-2003
		AT261146T T	15-03-2004
		AU763539B B2	24-07-2003
		AU773702B B2	03-06-2004
		AU2652999 A	23-08-1999
		AU4995299 A	07-02-2000
		BR9912001 A	04-12-2001
		CA2320261 A1	12-08-1999
		CA2337224 A1	27-01-2000
		CN1197027C C	13-04-2005
		DE1125219T T1	23-05-2002
		DE69909839D D1	28-08-2003
		DE69909839T T2	27-05-2004
		DE69915333D D1	08-04-2004
		DE69915333T T2	07-04-2005
		EP1053524 A1	22-11-2000
		EP1125219 A1	22-08-2001
		EP1143337 A1	10-10-2001
		EP1422640 A2	26-05-2004
		ES2206132T T3	16-05-2004
		ES2221404T T3	16-12-2004
		HK1041328 A1	20-08-2004
		IL140793D D0	10-02-2002
		JP2002503001T T	29-01-2002
		JP2002520735T T	09-07-2002
		NO20004010 A	10-10-2000
		US6108703 A	22-08-2000
		US6185598 B1	06-02-2001
		US6553413 B1	22-04-2003
		US7054935 B2	30-05-2006
		US7103645 B2	05-09-2006
		US2001056500 A1	27-12-2001
		US2005198334 A1	08-09-2005
		US2006218265 A1	28-09-2006
		WO0004458 A1	27-01-2000
		WO9940514 A1	12-08-1999

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CA2006/000434

WO2004079497	16-09-2004	AU2004217318 A1	16-09-2004
		CA2516975 A1	16-09-2004
		EP1625466 A2	15-02-2006
		US2005021999 A1	27-01-2005
US2002002636		AU5163601 A	30-10-2001
		AU5164301 A	30-10-2001
		AU5164401 A	30-10-2001
		AU5353201 A	30-10-2001
		AU5353301 A	30-10-2001
		AU5353401 A	30-10-2001
		AU5353601 A	30-10-2001
		AU5353701 A	30-10-2001
		AU5355901 A	30-10-2001
		AU5361301 A	30-10-2001
		AU5544101 A	30-10-2001
		AU5705801 A	30-10-2001
		AU5907401 A	30-10-2001
		AU5907501 A	30-10-2001
		US6990531 B2	24-01-2006
		US7020783 B2	28-03-2006
		US7043563 B2	09-05-2006
		US7111006 B2	19-09-2006
		US7120662 B2	10-10-2006
		US2002002602 A1	03-01-2002
		US2002002603 A1	03-01-2002
		US2002002611 A1	03-01-2002
		US2002002625 A1	03-01-2002
		US2002004816 A1	10-01-2002
		US2002007404 A1	17-01-2002
		US2002023159 A1	21-02-2002
		US2002056006 A1	09-05-2002
		US2002059170 A1	16-05-2002
		US2006129697 A1	15-06-2006
		WO0180002 A1	25-10-2001
		WO0180003 A2	25-10-2001
		WO0180004 A2	25-10-2001
		WO0180014 A2	25-10-2001
		WO0180024 A2	25-10-2001
		WO0180033 A2	25-10-2001
		WO0180062 A2	25-10-2001
		WO0180063 A2	25-10-2001
		WO0180064 A2	25-10-2001
		WO0180093 A2	25-10-2001