Securing the Anonymity of Content Providers in the World Wide Web

Thomas Demuth and Andreas Rieke

Department of Communication Systems

FernUniversität Hagen

Germany

ABSTRACT

Nowadays the World Wide Web (WWW) is an established service used by people all over the world. Most of them do not recognize the fact that they reveal plenty of information about themselves or their affiliation and computer equipment to the providers of web pages they connect to. As a result, a lot of services offer users to access web pages unrecognized or without risk of being backtracked, respectively. This kind of anonymity is called user or client anonymity. But on the other hand, an equivalent protection for content providers does not exist, although this feature is desirable for many situations in which the identity of a publisher or content provider shall be hidden. We call this property server anonymity. We will introduce the first system with the primary target to offer anonymity for providers of information in the WWW. Beside this property, it provides also client anonymity. Based on David Chaum's idea of mixes and in relation to the context of the WWW, we explain the term "server anonymity" motivating the system JANUS which offers both client and server anonymity.

Keywords: Anonymity, Privacy, World Wide Web, Server Anonymity, Client Anonymity, Encryption

1. INTRODUCTION

In recent years the usage of the WWW increased proportionally concerning its meaning and the number of users. It developed from a service focused on academic areas offering scientific content into a medium for providers of information of very different kind and sometimes doubtful seriousness.

So the request for anonymity in the WWW is immense; based on the fact that a user in the WWW gives away plenty of personal information while navigating through the WWW, it is possible to build a very personal profile of this user. The accumulation and connection of this information contradicts the usual idea of data security, violates personal rights, and offers many illegal possibilities like insertion into unwanted address lists which often results in undesired advertisement or spying out personal tendencies.

Because of this situation, services have already been built offering web users to stay anonymous. The development of appropriate mechanisms to protect anonymity is part of the research work in cryptology. Besides the protection of users, the protection of the service provider is coming up. The need for the establishment is manifold, four example situations are given below:

- A researcher wants to contribute a paper to a conference but is asked to do this in an anonymous way. This may be easy for the author's name and affiliation, but what about references to web documents (URLs)?
- A company wants to make a survey about a new product. This product should not be associated with that company in order not to influence people that participate in the survey. If the survey can be published anonymously and does not contain hints about the company it will come to a representative result.
- A similar situation would be the survey of a product of a competitive company. If the surveying company would do it using its name the rival firm would become attentive.

email: {thomas.demuth, andreas.rieke}@fernuni-hagen.de

- An electronic blackboard contains the advertisements of employees wishing to change their company. It enables the employees to present their proficiency anonymously by referring to a web page where their skills are listed. This web page may even be localized on the web server of the company the employees are actually working for. Thereby the employees do not have to fear about disadvantages by their employers.
- In the Yugoslav civil war the Internet was one of the few possibilities for civil right groups to communicate with the world. They were able to report about the injustice from their government by email and newsgroups but not by WWW because of the danger of backtracking. There and in other totalitarian countries it would be possible to realize the anonymity of speech (n. b.: the authors of the US constitution first published it under the name "Anonymous"!).

In general the anonymity of the sender, here the server, is of interest in case the relation shall be hidden for the client, or for a third party. A concept and the implementation of the prototype of a mechanism solving this problem of bilateral protection is presented in this article.

The paper is organized as follows: In section 2, we show briefly the concept of mixes introduced by David L. Chaum. In section 3, we explain the term "server anonymity" generally and more detailed in the context of WWW motivating the system JANUS. The method how JANUS works is described in section 4 in a general and in section 5 in a more formal and detailed way. Since there is the possibility of misuse in the nature of this system providing anonymity for a server, ways to solve these problems are presented in section 6. Section 7 describes the technical background of the implementation of our prototype and the availability of JANUS in the WWW. The paper ends with a summary in section 8.

2. MIXES

In his article[1] from 1981, David L. Chaum described among other things a general purpose mail system that conceals the relation between senders and receivers of messages in a system from a global aggressor. Messages are transferred through several intermediate stations called mixes. Each mix is able to hide the content of a message by securing it with a virtual envelope consisting of public key encryption (often RSA, but this kind of encryption is slowing down the whole processing). Furthermore, a mix can delay the delivery, change the sequence of arriving messages, or change their length to conceal the origin. In addition a mix is able to generate and transfer dummy messages in case too few messages are available. This guarantees that messages leave the mix at fixed time. In mix compounds, it is too expensive for a message to pass each mix, therefore a route is calculated through the network for each message in advance.

Further, Chaum describes a technique to veil the identity of the receiver of a message. Therefore the address is structured in a way that the receiver cannot be recognized by inspecting the address; details are shown in section 5. Chaum uses the term "untraceable return addresses"; but these addresses can also be designated as "anonymous return addresses" within the meaning of unlinkability of such an untraceable address and the clear text address of the belonging receiver. Mixes on a route through a mix compound are able to analyze an untraceable address step by step.

The system described in this paper uses untraceable return addresses to make references in WWW documents unrecognizable. In the following section, we will deal with the aspect of server anonymity in a more detailed form.

3. SERVER ANONYMITY

Existing projects and implementations investigate anonymity in various degrees, but only for the client. This article will extend the nature of the problem to the view of an information provider and his anonymity (server anonymity). There is both the desire and the demand for participants in the WWW to publish information without revealing the server's address. We will now describe the mode of operation of the protocol underlying the WWW to show the starting points to obtain server anonymity.

The WWW presents itself as a giant hypertext document consisting of separate globally distributed documents. Most of these documents are composed of textually and/or graphically oriented content and address information; from now on we will assume this as standard case. But this assumption and our research results can be adapted to other document structures used in the WWW.

The URL (Uniform Resource Locator) is the central and connecting element in documents. The URL makes it possible to identify and localize a document in a unique way. However, this URL also reveals information about the origin of a document, because it is often built according to the following syntax:

```
[scheme]://[server].[domain]/[path on the server]/[document]
```

Each of these components reveals more or less information about the author of a document; usually the domain section represents the most sensitive element.

Even if there is no clue from the domain to the geographical locality or the membership to an institution or organization of the server, there are many more mechanisms to extract detailed facts from this restricted information. Examples are global or national institutions like NIC (USA) or DE-NIC (Germany) whose tasks are to administrate the relation between domain name and the organization to which it belongs.

The HTTP (HyperText Transfer Protocol) describes the syntax of an URL, the mechanism of communication between web server and browser, but also the structure of messages exchanged between them. The specification of HTTP version 1.0, which is supported by our system, described in section 4, can be found online in the WWW[2] (about restrictions of this support see section 7). The communication between web browser and web server works in a bilateral way. The browser initiates a request consisting of two parts:

1. Header: The header contains meta information and data fields specifying the address to be contacted. It may contain the client's email address, the type of browser the client uses, and other information.
2. Body: The actual content (e. g. parameters for forms) is transported in this part.

After receiving a request the web server reacts with a response which is constructed similarly, but contains information different to the request; now, the content may be either the requested data or an error message of the server. Therefore, one has to remember that there is data in the header giving information about the server. The content of the message, representing web pages or other web objects, contains similar information to that in the header. The body of the message often includes web pages built with the language HTML (HyperText Markup Language); this language is related to the SGML-standard (see documentation about HTML version 3.2[3]). HTML is the most often used language to construct hypertext documents in the WWW. HTML consists of instructions (tags) and the actual information. Some of these tags contain references to other objects in the WWW. Because of the references' nature of being address information they have to be treated specially by the described system.

Considering these facts, it would be an ideal situation for a content provider to publish the URLs of his web pages in a way that

- on the one hand does not allow to backtrack directly and
- allows the use of a standard web browser on the other hand.

The system JANUS described in the following fulfills these requirements: It represents a network of stations working with mix methods. The partners of a communication stay anonymous. That means that the identity of the content provider stays hidden to the user in the situation described above. The transport of the messages takes a route over one or more instances to increase security.

JANUS is an acronym for "Justly Anonymizing Numerous URLs Systematically", describing the features of the system:

- Justly: The authors assume that the anonymous appearance of web servers is legitimate in general, exceptions are described in section 6.
- Anonymizing: The central function of the system is anonymity for both client and server.
- Numerous: Not only the references in the header but also those in the content are encrypted, adding up to numerous encryptions per request.

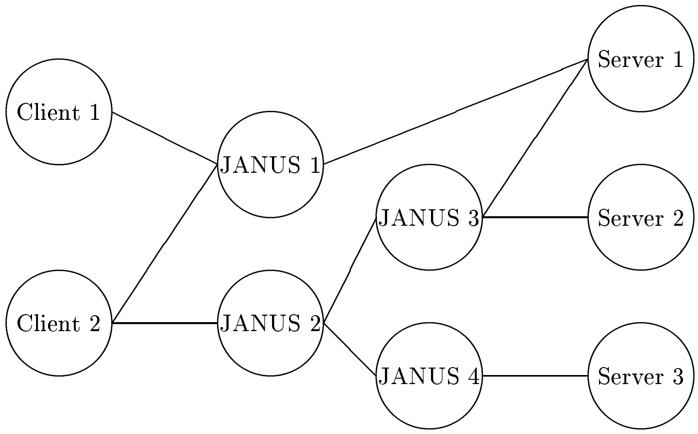

Figure 1. A possible JANUS network

- URLs: The central objects of the system.
- Systematically: We use uniform cryptologic methods for encryption and decryption.

But JANUS can also be associated with the homonymic roman god with two faces. His task was to observe portals and bows; the system JANUS' task is to investigate passing information. Because a portal possesses two sides, JANUS provides anonymity in both ways, for the server as well as for the client.

4. METHOD OF WORKING

Figure 1 shows a possible constellation of web clients, web servers, and JANUS servers. One can see some web clients (clients 1 and 2) and their partners, the web servers 1 to 3. Some instances of JANUS are located between the content providing servers and the clients. The communication is not restricted to one instance of JANUS (JANUS 1); it is possible to use cascaded JANUS systems to improve security and immunize the system against attacks in form of eavesdropping of incoming and outgoing messages (JANUS 2 to 4).

An approved method for the encryption of data, in this case of web addresses, are public key algorithms. There are two alternatives to publish an URL anonymous:

1. The provider uses the actual function of JANUS, a special web page or
2. he encrypts the URL with the public key, published by the operators of the JANUS server.

With a symmetric algorithm it would not be possible to publish the secret (and only) key, because everybody using our system would be able to decrypt URLs encrypted by another user.

These encrypted URLs are submitted to users who will use them to contact the web page via JANUS. If this encrypted URL is sent from a web browser to JANUS, JANUS decrypts it with the secret key. After that JANUS acts like a web browser itself and requests the web page the URL is pointing to. The received page is being analyzed for URLs to be encrypted, header information is anonymized or filtered, respectively; the resulting page is sent to the client.

We have to pay attention to the fact that there is not only content in a request or response. Administrative data is transmitted in the header of the message, too, and this data has to be treated in the same way to achieve anonymity. Our method is a kind of partial realization of Chaum's concept. But our encryption relates to possible revealing references instead of the whole content. This point is treated in section 5.

This means, that a content provider who wants to be anonymous, has to publish the URL of a web page in encrypted form. He is responsible himself not to reveal his identity by compromising content.

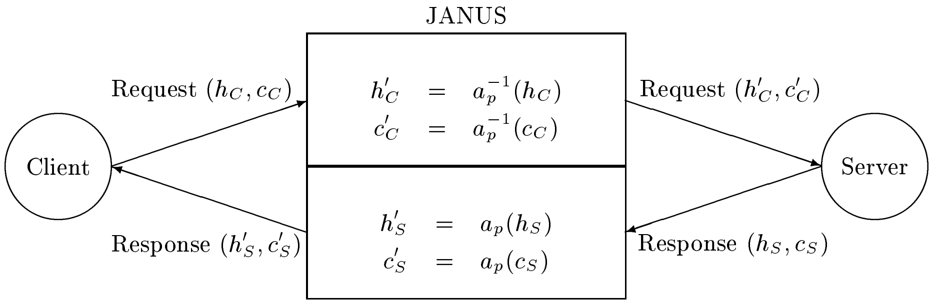

Figure 2. JANUS working in context "encryption"

5. FORMAL DESCRIPTION

In the following, we will explain JANUS' central functions: anonymization and encryption. Figure 2 illustrates the method of working for the prefix (the context JANUS works in) "encryption" (JANUS can work in different contexts, dependent on the used prefix; prefixes and their meaning are listed in table 1):

Field name	Handling
janus_encrypted/	Takes an encrypted URL as argument
janus_encoded/	Takes an encoded URL as argument
show_request/	Shows the HTTP-request in clear
show_response/	Shows the response to the given HTTP-request in clear
encrypt_url.html	The given URL is encrypted and displayed
encode_url.html	The given URL is encoded and displayed

Table 1. Prefixes and their meaning

In preparation we will now define some functions:

Detection function for URLs in the input stream:

$$U : S \to U, \text{ with } I \text{ input(stream)}, U \text{ set of URLs}$$

$$U(x) = \text{ Set of all URLs in } x$$

Anonymity function a_p and its complement a_p^{-1}

$$a_p : S \to S, \text{ with } I \text{ input(stream)} O \text{ ouput(stream)}$$

$$a_p(x) = \begin{cases} u_i \to p(k_e(u_i)): & \text{for all } u_i \text{ in } U(x) \\ e \to \epsilon^*: & \text{for not supported language elements } e \text{ in } x \\ \text{unmodified:} & \text{in any other case} \end{cases}$$

$$a_p^{-1}(x) = \begin{cases} u_i \to k_d(p^{-1}(u_i)): & \text{for all } u_i \text{ in } U(x) \\ e \to \epsilon: & \text{for not supported lang. elements } e \text{ in } x \\ \text{unmodified:} & \text{in any other case} \end{cases}$$

k_e **and** k_d**:** Functions of the public and secret key of the used public key algorithm.

* ϵ : empty set

p and p^{-1}: Prefix function and its inverse.

The task of the anonymity function a_p and its complement is to encrypt and decrypt references considering a context and to filter elements out of the content not supported by JANUS or to modify them in a way that they become harmless. This can be information about the email address or the type of the used web browser. Further elements of the input stream stay untouched.

In the following we will investigate header and content of a message separately; on both a_p and a_p^{-1} are working formally the same, but we have to keep in mind that the header's structure is constant according to the underlying protocol, the construction of the content can be different dependent on the transmitted message (see comment below about functions working specifically in relation to the content).

The user knows an URL $url_{encrypted}$, encrypted with the public key k_e, and concatenated with a prefix p. With a standard web browser (client), he sends JANUS a request, consisting of a tuple of header and content (h_C, c_C). JANUS then calculates a modified header/content tuple:

$$h'_C = a_p^{-1}(h_C)$$

$$c'_C = a_p^{-1}(c_C)$$

As a side effect, the prefix function determines the further specific behaviour of JANUS from the input. In the following we will see the behaviour if the prefix describes an encrypted URL. JANUS removes the prefix from the relating header field of the response and decrypts via k_d:

$$url = url_{decrypted} = k_d(p^{-1}(url_{encrypted})) = k_d(p^{-1}(p(k_e(url))))$$

At this point JANUS knows the unencrypted URL url and constructs a request (h'_C, c'_C) for the web server implicitly addressed. This web server receives the request and answers with a tuple (h_S, c_S) which is sent back to JANUS.

The treatment of this response corresponds to the one of the requests, but with regard to the inverse anonymizing function a_p:

$$h'_S = a_p(h_S)$$

$$c'_S = a_p(c_S)$$

The function a_p encrypts all relevant references in header and content:

$$url_{encrypted} = p(k_e(url_{decrypted})) = p(k_e(k_d(p^{-1}(url))))$$

The pair (h'_S, c'_S) is sent back to the client as response.

We assume that the content c consists of text and address information built with the HyperText Markup Language described in section 3. Each address (URL) in this text exists in c'_S in encrypted form. The function $U(x)$ delivers this set of references in HTML documents and can be adapted to other document types; adapted working functions for formats like adobe acrobat, java applets, and others used in web pages are possible. Starting with an encrypted URL the client gets only encrypted references in further requests because of the recursive nature of our method.

The anonymity functions a_p and a_p^{-1}: Public key cryptosystems are Chaum's choice for encryption and decryption. The public key can be spread without revealing the secret one. A publisher can encrypt an URL on his own, too, and does not have to use JANUS' function for this. The real enciphering of submitted URLs is done by the anonymity function a_p using the public key functions k_e.

The prefix function: JANUS receives an URL relevant to it (starting with `"http://janus.fernuni-hagen.de/"`) and splits this URL into

- this starting string, which is not relevant further,
- a prefix, which determines in which context JANUS handles the next argument, and the
- real URL to be handled (this URL may be encrypted).

The prefix determines actions like

- decryption of the argument (the passed URL),
- a presentation of a request in clear text, and
- the presentation of the response of a contacted server in clear text.

The prefixes supported by JANUS in version 1.0 are listed in table 1.

6. RESTRICTIONS AND MISUSE

A service that is publicly available making it possible for a content provider to stay anonymous may cause misuse. Because of this fact, JANUS is limited to free accessible web pages only in this first version (1.0). Therefore, it is more difficult to use JANUS for such content to make profit of in a idealistic or material way. JANUS is not available for providers of web pages containing information that offend against German, European or international law or violate common moral standards. Those providers will be inserted in a restriction list. Entries in this list have the effect that a user will get a page with a hint from JANUS instead of the real addressed web page, if he wants to access this page. The URL of offending web pages will be added to the list if the authors will be noticed of a case of JANUS' misuse stated above. In addition the decrypted URL of this provider can be notified to prosecuting authorities on demand after the proof of misuse.

The main reason for not encrypting the content of messages by JANUS is the maintenance of legitimacy. Otherwise there would be no chance for the authors to control and restrict the use of JANUS for certain users.

7. IMPLEMENTATION

At the FernUniversität Hagen and the FTK (Forschungsinstitut für Telekommunikation - Research Institute for Telecommunication) a prototype (version 1.0) of the system JANUS has been implemented. In addition to the protocol HTTP and HTTPS, JANUS is able to handle the protocols FTP and GOPHER in this version.

7.1. Programming details

JANUS has been written in the script language PERL (version 5.004_04). For the development of WWW server and client functions, for the handling of HTTP and HTML, and for the encryption we have used perl libraries (libwww-perl 5.14, cryptix 1.16).[4]

7.2. Platform

At this time JANUS is running on a SUN Sparc Ultra with two CPUs.

7.3. Encryption

For encryption and decryption JANUS uses the public key algorithm RSA.[5] A modulus of 768 bit is used in version 1.0. Because of speed reasons for encryption, we use the Fermat number F_4 ($2^{16} + 1 = 65537$) as public key. When using JANUS to access web pages URLs are encrypted rather than decrypted in the majority of requests. Using F_4 the calculation can be speeded up.

The encryption function gets the URL as text input. This URL is converted into a long integer on ASCII base, split into parts according to the size of the modulus if needed. The result is encrypted with RSA and sent through a modified BASE64 algorithm[6] because an URL has to contain only characters of a specified set.

The modulus and public key of the used RSA algorithm is accessible in public. Therefore, an encryption of an URL does not need the function of JANUS.

7.4. Intruder Model

In our prototype JANUS does not support the encryption of content and differs therefore from Chaum's concept. The prototype uses a modified intruder model: There are two areas of trust on the way information flows from a client through some JANUS to a server and vice versa. These are the connections between the server and the first JANUS and the last JANUS and the client, respectively. A content provider has to trust this first JANUS. If an attacker is not able to tap these connections, he can not uncover the relation of communication between client and server until he shortcuts all JANUS outside the areas of trust.

7.5. Secure publishing

JANUS guarantees anonymity for a content provider under the condition, that this provider does not reveal information about himself through the content of his web pages. Therefore, he has to care about it before publishing; he has to avoid textual hints or URLs and email addresses in the text. JANUS anonymizes references but does not modify further content. The access to pages published in this way has to be denied for search machines (robots) in WWW because these machines are able to find the unencrypted address of the page while searching for its content. By reorganizing a web server, web links often become inconsistent. If such a site is requested a web server will react normally with a notice like "File xyz.html not found", showing the filename or even the name of the server in clear. JANUS is able to handle this publisher's error and reacts with an own web page not showing the filename in clear.

The web server of a content provider has to be configured correctly. It should never answer with a directory listing if the file name in an URL is omitted.

7.6. Restrictions

JANUS does not support Java, JavaScript, or Cookies in version 1.0. It let pass all HTTP/1.0 header fields an removes all other fields. Exceptions from this rule are given in table 2.

Field name	Handling
HTTP/1.0 header fields:	
From	Contains the clients email address and is thus removed by JANUS.
Location	Contains the new URL in case of redirects and is thus encrypted by JANUS.
Pragma	Is not exactly specified and thus removed except in case of the no-cache directive.
Referer	Contains the referring page and is removed by JANUS since that page may contain information about the client.
Server	Contains information about the servers software and is thus removed.
User-Agent	Contains information about the clients software and is thus removed.
WWW-Authenticate	Is needed in case of access to protected web pages and is thus removed.
Authorization	Is needed in case of access to protected web pages and is thus removed.
HTTP/1.1 header fields:	
Accept-Language	Specifies the languages the client is willing to accept. Although this reveals information about the client, it is not removed since it enables a useful functionality.
Content-Language	Specifies the language of the content. May be used to choose an appropriate language and does not reveal any information since the contents language is obvious.
Host	Contains the servers hostname and is often necessary to select the correct virtual host. This field is changed by JANUS.

Table 2. Handling of HTTP header fields

7.7. JANUS in the WWW

JANUS is accessible under `http://janus.fernuni-hagen.de/` and `https://janus.fernuni-hagen.de/` in the WWW. The second URL provides client oriented security using the SSL (Secure Socket Layer) protocol. All of JANUS' special functions are accessible over this web page, their use is explained, too. One of JANUS' central functions is the possibility to encrypt an URL to publish the enciphered version. There are FAQs and analyzing functions for HTTP requests and responses.

8. SUMMARY

In this article the authors have explained and motivated the meaning of the term "server anonymity". We have described a system helping to guarantee this kind of anonymity and to make it possible to publish anonymously in the WWW. The potentialities of misuse with this system and their prevention are discussed.

Based on the concept of mixes by David Chaum, there are some differences between our prototype and his concept. These are encryption of content, creation of dummy information flow, uniform length of messages, or the order of messages. We will take this points in consideration in our newest research activities and therefore in a paper to appear.

The JANUS prototype started operation in November 1997 and receives up to 7000 requests per day. In addition we did not receive any hints of JANUS' misuse.

ACKNOWLEDGMENTS

We wish to thank the Forschungsinstitut für Telekommunikation (FTK - Research Institute for Telecommunication), the department of Communication Systems at the FernUniversität Hagen under supervision of Prof. Dr.-Ing. Firoz Kaderali, and our colleagues for their support in our research. We are also grateful for the advice of Prof. Andreas Pfitzmann and his colleagues from the Universität Dresden, Germany.

REFERENCES

1. D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms", *Communications of the ACM* **24**, pp. 84–88, February 1981.
2. "http://ds.internic.net/rfc/rfc1945.txt"
3. "http://www.w3.org/markup/wilbur/"
4. "http://www.systemics.com/software/cryptix-java/"
5. R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM* **21**, pp. 120–126, February 1978.
6. N. Borenstein and N. Freed, "Mime (multipurpose internet mail extensions): Mechanism for specifying and describing the format of internet message bodies", RFC1521, Sep. 1993.