A Passive Attack on the Privacy of Web Users Using Standard Log Information

Thomas Demuth

Department of Communication Systems*
University of Hagen
Germany
thomas.demuth@fernuni-hagen.de

Abstract. Several active attacks on user privacy in the World Wide Web using cookies or active elements (Java, Javascript, ActiveX) are known. One goal is to identify a user in consecutive Internet session to track and to profile him (such a profile can be extended by personal information if available).
In this paper, a passive attack is presented that uses information of a different network layer in the first place. It is exposed how expressive the data of the HyperText Transfer Protocol (HTTP) can be with respect to identify computers (and therefore their users). An algorithm to reidentify computers using dynamically assigned IP addresses with a certain degree of assurance is introduced. Thereafter simple countermeasures are demonstrated.
The motivation for this attack is to show the capability of passive privacy attacks using Web server log files and to propagate the use of anonymising techniques for Web users.

Keywords: privacy, anonymity, user tracking and profiling, World Wide Web, server logs

1 Introduction

The necessity for the protection of user privacy on the Internet is hard to understand for the normal user in general; their opinion is that ,,there is nothing to hide". The majority of the average Internet users are using the World Wide Web (short: WWW) as their main Internet service[1] without being conscious of the fact that they give away plenty of information while browsing. This information is given both implicitly and explicitly:

– Protocol information
  While communicating, the Web browser of the user and the Web server of a

* The author would like to thank the Department of Communication Systems under supervision of Prof. Dr.-Ing. Firoz Kaderali, the colleagues at the department, and especially Prof. Dr. rer. nat. Werner Poguntke for the support in his research.

[1] Many of them even equate Internet and World Wide Web.

content provider are exchanging data via the HyperText Transfer Protocol (HTTP) [1]. Besides the information necessary for accessing a Web resource (Web page, video clip, mp3 file, ...) additional information is transferred to influence the reaction of the server (e.g. choice of the preferred resource language, compression type, etc.). This data is transferred within the HTTP headers (see Section 2) and can be very expressive.

– Personal information
  Using the Internet means searching for information (in most cases). From time to time, every user also gives away some of his personal data voluntarily (surveys, etc.).

To trace users over a long time and consecutive Internet sessions, the IP address of the user's computer is often used and therefore logged (besides some protocol information). Each IP address, identifying a computer uniquely on the Internet, can be used to reidentify a computer (and therefore its owner).

Most Internet users do not have a dedicated Internet connection but dial in via modem to their Internet Service Provider (ISP). In general, those ISP have less IP addresses available than they have customers. Hence, the customer's IP address out of a pool of available addresses is assigned dynamically at the time of dialing in; this assignment is therefore quasi-random. The probability that the same IP address is assigned to the same customer the next time he dials in depends on the ISP's number of customers and the size of the IP address pool. Only the ISP is able to reassign an IP address to a customer after his Internet session has ended.

It is often heard that privacy of these users is sufficiently secured against third parties interested in profiling [2, 3].

In the following it is shown, that this opinion is a misjudgement and a passive attack against the privacy of Web users with dynamically assigned IP addresses is presented based only on protocol data that can be logged by every common Web server.

The goal of this attack is to identify users with dynamically assigned IP addresses between different Internet sessions.

The motivation for this attack is to show the necessity of the use of anonymising techniques even for the normal Web user.

2 The HyperText Transfer Protocol

Web browser and Web server are communicating using a standardised protocol, the HyperText Transfer Protocol (HTTP) [1]. Each request for a Web resource issued by a Web browser and addressed via a so called Uniform Resource Locator (URL) is answered with a response by the server. The server is normally waiting for requests at port 80; the requests consist of normal ASCII text, can therefore easily be read by humans, and are structured into fields. Besides the type of access to a Web resource (`GET, PUT, POST, ...`) the URL and an arbitrary number of fields are transferred. These fields contain a name and corresponding

content. As an example, Table 1 shows the HTTP header fields of a request for the resource at URL `http://www.amazon.com/` using an Opera Web browser.

Table 1. HTTP header content for a request to `http://www.amazon.com/`

```
GET http://www.amazon.com/
Cache-Control: no-cache
Connection: Keep-Alive
Pragma: no-cache
Accept: text/html, image/png, image/jpeg, image/gif, image/x-xbitmap, */*
Accept-Encoding: deflate, gzip, x-gzip, identity, *;q=0
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Host: www.amazon.com
User-Agent: Opera/5.0 (Linux 2.2.16 i686; U)  [en]
```

The meaning of the the most important and most often used header fields is explained and commented in Table 2.

Table 2. Most often used HTTP header fields

HTTP/1.1 header fields:	
Field name	Meaning
User-Agent	Contains information about the client's configuration
From	Contains the clients's email address
Accept	The accepted media types (also an index to installed software)
Accept-Language	Specifies the languages the client is willing to accept
Accept-Encoding	The accepted encoding (compression type, etc.)
Accept-Charset	What character sets are acceptable for the response
Method	{OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT}
Host	Contains the server's hostname
Via	Can contain a specification from the proxy which was used to transfer the request
If-*	A set of fields for conditional access to resources (with reliance on the resource's time of creation, modification, etc.)
Besides the information above also the type of Server-Protocol (HTTP/1.0 or HTTP/1.1) is transferred	

The content of the fields varies from browser to browser and depends on the browser's underlying operating system, language preferences set by the user of a browser, software installed on the user's computer, and many more individual conditions.

Each request for a resource is normally logged by the Web server. Most Web servers are configured to do that by default and most of the server administrators are not interested in changing this behaviour due to lack of interest or competence. Organisations caring about the privacy of Internet users like *Privacy Watch* (and also federal privacy officers) criticise this behaviour.

Logging data consists of the URL, date, time, and the most often used HTTP header fields in general. To enhance the amount of logged data to all HTTP header fields only little configuration of the Web server is necessary.

3 Problems of Passive Attacks

Passive attacks are based on analysing logged data. This data is very easily available. Even if an attacker is not owner of a Web server but only using hosting services of an ISP, an attack is easy to perform: Nearly all ISP send the recorded access data to their customer periodically. This log data does not only contain the accesses to the customer's Web pages but in most cases also statistics about time, geographical region, and domain of the Web page's visitors.

Focus of a passive attack is the IP address of a user's computer. If this address is static and therefore does not change over time, it is very simple to identify and trace a person.

But the success of this method is restricted: The address space of the current Internet Protocol is limited. In earlier years it was not unusual for a customer of an ISP to get a static IP address. But more and more IP addresses are assigned dynamically by ISPs. In most cases, the Internet Protocol Control Protocol (IPCP, [4]) (used by the Point-To-Point-Protocol, PPP) is used to handle the process of dial in at the provider; PPP can also perform the assignment of IP addresses to the client. Each time a customer dials in, a currently unused IP address is taken from the pool of addresses belonging to the provider; this process is nearly pseudo random.
Because some ISPs have more than a million customers (like AOL in the United States or T-Online in Germany) the number of IP addresses available does not belong to a contiguous range. Furthermore, even IP addresses assigned in two consecutive Internet sessions do not have to match in any position.

As a conclusion of the former explanations it is obvious that user tracking by IP address is futile.

4 Tracking Web Users by HTTP Headers

The protocol information of another OSI layer could also be used to track users. Regarding the most often used Internet service, this would be HTTP. While IP addresses can change from session to session, some HTTP header fields have to stay unchanged to warrant the intended operation and communication between Web client and Web server. But even if relying on a certain set of mostly invariant HTTP headers, slight modifications of browser configurations have to

be tolerated. This is important because users modify the browser settings directly and installed or removed software can have an effect on HTTP headers alternatively or additionally.

A successful attack on the privacy of a Web user can be performed as follows: Given a line of server log data not mandatory containing IP addresses and describing one access to a Web page by a person using a Web browser and a set of such protocol lines, possibly containing other accesses by this user, these accesses can be identified with a certain degree of reliability. The line of log data consist of some meta information (date and time of the access) and the content of a set of chosen HTTP header fields.

4.1 Terminology

The term *Access Data Set (ADS)* shall describe a set of

- a timestamp ts describing date and time of access to an URL and
- a set of terms $t_{1,1}, \ldots, t_{m,n}$, the contents of a selected number of HTTP header fields $\{h_1, \ldots, h_m\}$. The number of terms a header field can contain depends on the type of the header field.

An *extended Access Data Set (eADS)* extends an ADS by additional personal information (e.g. given to a Web form) by the corresponding user. In realiter it is interesting for an attacker, possessing an eADS (and a great number of accesses to different URLs on different Web servers) to (re)identify a user to connect this eADS to other (e)ADS. This gives him the opportunity not only to track the user but also to profile him. Depending on the computer performing even real time tracking is possible.

An *instance* synonymously means a person using a Web browser to access Web pages and likewise this particular Web browser implicitly defined by its configuration.

Instances *generate* (e)ADS while accessing Web pages.

4.2 Identifying Sensitive HTTP Information

The goal of the attack is to identify instances by their ADS (and therefore by HTTP headers). Which HTTP fields are the most expressive and are worth to be considered nearer?

Type 1: As mentioned, some header fields are intended for transporting instances between server and client only and can not be seen at the end points of the communication.

Type 2: Other header fields can only contain a few different terms like the `HTTP-Version` field, which can only include "HTTP/1.0" or "HTTP/1.1", or the `Method` field, which can contain an entry of the set {`OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT`}. An analysis of the ADS base used to verify the attack in Section 4.5 showed, that in more than 99% of the ADS only an element of {`GET, POST`} is used.

In general, the more terms a header field can contain the more individual and expressive it can be: A header field that can span over up to p terms out of n possible terms with the restriction, that every term can be included at most once, can "mark" $\binom{n}{p}$ ADS individually.

Type 3: A deeper analysis can be used to identify a number of header fields much more individual. These are fields like `User-Agent` or `Accept`. These fields are very invariable over a long time (at least until an instance changes its operating system or the browser type).

Table 3 shows the results of an analysis of HTTP header fields used in the attack based on an ADS set described in Section 4.5: the number of different terms in each field is listed.

Table 3. Used HTTP headers and number of different terms in the ADS base

Field name	Term count	Field name	Term count
Host	459	Trailer	1
User-Agent	318	Warning	1
Server-Protocol	5	Via	1248
Accept	32	Range	19
Accept-Language	159	If-Range	17
Accept-Encoding	7	If-Match	2
Accept-Charset	5	If-None-Match	21272
Method	4	If-Modified-Since	1
Expect	0	If-Unmodified-Since	2
From	10	**Sum**	**23562**

Definition: A *frequency of a term* is defined as the quotient of the number of occurences of this term and the number of all considered terms.

Even if a header field can contain one or more of a high number of terms, a term's frequency in a set of ADS is determining the significance of this term and the significance of an ADS containing this term. A field containing terms evenly distributed is only as significant as a field of type 2.

ADS of the same instance vary over time: New software is installed which can have impact on the `Accept` field, a proxy can be used (which modifies the `Via` header and can extent the `User-Agent` field), or the language settings are modified which effects the `Accept-Language` field, etc.

To trace the Web page accesses of instances, it is intended to determine correlations between different ADS. This situation can be equated with the problem to find documents in a document database. There, for a query formed of keywords, the most relevant results shall be found. This problem has been solved in the field of *Information Retrieval*.

4.3 Using Methods of Information Retrieval

Information Retrieval is the field of searching and finding of information in large data pools. Not every attribute or bit of information is relevant for searching; because of redundancy in data objects not the objects themselves but representations are stored in a database, the objects have to be filtered. These representations must describe the data objects as good as possible. On the other hand, queries to the database must be formed equally to fit for the representations, a transformation function is necassary for both requirements.

In our attack, the data objects are the ADS an adversary has collected, the query is represented by an (e)ADS of an instance that shall be (re)identified and tracked.
The filter is identical to the consideration of only the relevant HTTP header fields as explained in Section 3. The transformation function is described in the following as part of the complete algorithm.

4.4 Preparations

The results of our simulated attack are based on an ADS set of 2.7 billion Web accesses. These are Web accesses to the same Web resource. In addition to the HTTP header fields, date, time, and IP address of the accessing client have been logged[2].

4.5 Algorithm

Analysis of the ADS Base The ADS base is analysed entry by entry, considering only the relevant header fields. The content of each header field h_j is parsed to identify the field terms $t_{j,1}, \ldots, t_{j,n}$. For each header field an ordered list of different terms found is kept. If a new term is found it is added to the header field term list, each term $t_{j,k}$ already known increments the counter $cnt(t_{j,k})$ of the correspondent entry in the list. The sum of all possible terms of a header field h_j is therefore $cnt(h_j) = \sum_{l=1}^{n} cnt(t_{j,l})$.

The function $l_j(a)$ is the number of terms in the header field h_j of a certain ADS a.

The significance of a term is identical to its inverse frequency in all data objects, in our model in all ADS. In correspondence to [5, 6] the so called *term weight* is:

$$weight(t_{j,k}) = -ld\left(\frac{cnt(t_{j,k})}{cnt(h_j)}\right)$$

The terms found are also called *index terms* because each ADS in the data pool can be described as a combination of these terms.

The difference of term weights considering one example header field (here: `User-Agent`) are graphically displayed in Figure 1. The impact of the term weight is obvious: The more common a term the less its weight.

[2] All users have been informed about the logging and its purpose for scientific research.

Fig. 1. Term weights of header field `User-Agent`

In addition, another measure has to be defined to estimate the weight of an ADS. It has to be considered, that ADS can consist of a different number of terms. To avoid, that a concise ADS with "heavy" but only a few terms has an equal weight as a common ADS, the weight of an ADS a is defined as follows:

$$weight'(a) = \sum_{j=1}^{n} \frac{\sum_{k=1}^{l_j(a)} weight(t_{j,k})}{l_j(a)}$$

with n being number of header fields of ADS a.

Construction of Index Vectors As mentioned, the Information Retrieval systems stores a representation of the data objects (here: the ADS), therefore a transformation function is necessary (as for the transformation of ADS to trace). Each ADS in the information system is represented by a binary *index vector* for two reasons:

1. More efficient storage
   Binary index vectors can be stored as sparse vectors. In our attack, the length of an index vector is 23623 where only 13 to 42 positions are set to 1.
2. Calculation speed up
   The determination of the similarity of two ADS can be sped up while using sparse vectors.

The length of each index vector l_{iv} is equal to the sum of all terms of all considered header fields. To build index vectors the term lists are concatenated: $(t_{1,1}, \ldots, t_{1,cnt(h_1)}, t_{2,1}, \ldots, t_{2,cnt(h_2)}, t_{3,1}, \ldots)$. The term at position j is called t_j.

An index vector of an ADS a is described as follows:

$$iv(a) = (b_1, \ldots, b_{l_{iv}}), b_j \in \{0, 1\} \text{ and}$$

$$b_j = \begin{cases} 1 & : \text{ term } t_j \text{ is member of } a \\ 0 & : \text{ else} \end{cases}$$

The value of index vector $iv(a)$ at position i is denoted $iv_i(a) = b_i$.

Search for Correlations For a given probe ADS a_{probe} of an instance to be (re)identified and tracked the similarity to each ADS a_i of the ADS base has to be determined. With reference to [5] the similarity of two data objects is equal to the scalar product of the corresponding index vectors considering the term

weight:

$$sim(a_{probe}, a_i) = \sum_{r=1}^{l_{iv}} \sum_{s=1}^{l_{iv}} iv_r(a_{probe}) * iv_s(a_i) * weight(t_r) * weight(t_s)$$

In our attack, a modification of the formula is necessary: Differences in terms are leading to a decrease of the similarity value by the term's weight (down to a value of zero, further differences do not lead to negative values). A similarity of 100% means identity of both ADS; in this case a_{probe} and a_i have the same weight[3].

This similarity (to the probe ADS) is calculated for each element of the ADS base.

Evaluation and Ranking A dynamically assigned IP address issued to an instance does not normally vary during an Internet session. For simplification, an instance is assumed, that is online once a day.
A lower bound of similarity has to be considered. Below that value ADS found are disregarded; their similarity to the probe ADS is too low. This threshold value is denominated by Δsim and is a relative value (ADS with similarity below $100 - \Delta sim$ are ignored). If an ADS a_1 is found whose similarity is above the threshold, it initialises a *potential activity period (PAP)*, ADS with a lower similarity can be neglected. This tolerance is necessary to eliminate variations in the ADS of instances over the time (as explained in Section 4.2). A PAP therefore forms a group of ADS assumed to be generated by the same instance at the same online session. Each consecutive ADS a_{found} found which fulfils the following criteria is added to this PAP P:

- The IP address is the same as the IP address of the initialising ADS.
- The similarity to a_{probe} is high enough (regarding to Δsim).

$$sim(a_{probe}, a_{found}) > 100 - \Delta sim$$

- The ADS lies within the time window given by the timestamp of the first ADS and the window size Δt.

$$\forall a_i \in P : ts(a_i) \geq ts(a_1) \wedge ts(a_i) < ts(a_1) + \Delta t$$

A PAP therefore is a representative for the potential occurrence of activity of an instance. Each PAP found by the presented algorithm consists of the initialising ADS and the ADS of the most and less similar ADS within the time window.

Any other ADS found which fulfills the similarity criterion but has a different IP address or does not lie within the time window initialises a new PAP.

[3] This relation is not symmetric: Two ADS with the same weight do not have to be identical.

Two PAPs with different IP addresses which are not disjoint regarding the time window build a *PAP intersection*[4].

It is obvious that the grade of anonymity is dependent on the number of PAP intersections of a probe ADS. The more intersections appear while calculating the PAPs, the more anonymous is the probe ADS, because each (but only at most one) of the found PAPs can be generated by the instance belonging to the probe ADS with the same probability. This property is a direct analogy to the mix concept developed by David Chaum [7], where each participant in a mix network can be the sender of a message with the same probability.

The more common the configuration of an instance, the more common are the ADS generated by it. But because of this property, there are also many instances that are generating such an ADS and therefore the probability of PAP intersections is very high for common instances, rsp. their configurations. The PAP intersections of a probe ADS are forming an *anonymity set* for the probe.

Furthermore, the following restriction is given:
An attacker can not be sure that a PAP found is generated by the tracked instance at all, because the combination of the terms in the header fields is not unique. Additionally, an attacker does not know the precision of the retrieval, because he can not distinguish between PAPs correctly identified and PAPs identified but not belonging to the traced instance.

4.6 Quality of Found PAPs

Anonymity is the state of being not identifiable within a set of subjects, the *anonymity set* [8]. In the present scenario of (re)identifying instances this set is the PAP found. Being anonymous, all members of this set can represent with the same probability the probe ADS. With the explained procedure some of the ADS out of the set are more likely identical to the probe ADS (depending on the determined similarity). Considering the restriction for eliminating consecutive appearances of instances (Δt) and the maximum allowed relative difference to the probe ADS (Δsim), one obtains the number of relevant PAP.

The grade of anonymity is equivalent to the number p of PAP intersections: With respect to the definition of the anonymity of a group of senders [9], the anonymity of the probe ADS is denoted as $ld(p)$ (the binary logarithm of the number of intersections).

The number of PAP intersections itself depends on several factors:

- The time window Δt
  The higher this value, the more intersections will be found. Because most users on the Internet are connected via dial in gateways (and therefore receive

[4] An additional feature of the attack is the potential to determine if an instance uses statically or dynamically assigned IP addresses (it is no mandatory assumption of the attack that the given instance to be tracked uses dynamically assigned IP addresses). If all other PAPs found present the same IP address, one can conclude that this address is static.

dynamically assigned IP addresses) they are online once a day. Thus, six to 24 hours are a appropriate value for Δsim.
- The similarity threshold Δsim
  Variations of several (hardware and software) configurations over some time are analysed. Evaluations have shown that the ADS weight varies up to 10.30% with respect to modifications mentioned in 4.2.

The restriction mentioned above is a factor influencing the assertion, that a found PAP, even if it is a member of an anonymity set of size one, is generated by the tracked instance. The attacker only knows the weight of an ADS, the number of PAP found, and the number of PAP intersections. Therefore he needs to use some empirical data.

4.7 Experiments and Results

To verify the correct function of the presented algorithm and to ascertain a measure for the quality of the found PAPs (and thus for the grade of anonymity), a modified ADS set has been created:
Nearly 300 ADS have been randomly created to build the probe set. Then these probe ADS have been "mutated" in a way a real instance would generate varying ADS (see Section 4.2) resulting in more than 13.000 probe ADS. Additionally, these mutated ADS have been marked so that they could be reidentified for verification, but without influencing the retrieval algorithm. The mutated ADS have been spread over the ADS base resulting in a modified ADS base.
In a second step the original probe set has been tested against the modified ADS base.

This way, the exact number of ADS to be found for each probe ADS is known and allows to judge the quality of the complete attack.

Two rates determine the accuracy of the retrieval (and therefore of the attack):

- Recall: a measure of how well the attack performs in finding relevant ADS. The recall rate is 100% when every relevant ADS is retrieved.
- Precision: a measure of how well the attack performs in not returning nonrelevant ADS. The precision rate is 100% when every ADS returned is relevant to the probe ADS.

With NRA (Number of relevant ADS), NAF (Number of ADS found), NIF (Number of irrelevant ADS found), and NRF (Number of relevant ADS found) the rates above are defined as follows:

$$recall = \frac{NRF}{NRA}$$

$$precision = \frac{NRF}{NRF + NIF}$$

Figure 2 shows the average recall and precision rates of the test set with the probe ADS for $\Delta sim \in \{0, 5, 10, \ldots, 100\}\%$. The attack is able to identify with an average recall rate up to 0.98 and with a precision rate up to 0.71. One can identify a local optimum at $\Delta sim = 35\%$. The attention has to be turned on a high value of precision, because every found PAP has to be stored and tracked step-by-step; PAP found with a precision lower than a given bound can be discarded. Therefore precision needs to be as high as possible.

Fig. 2. Recall, precision, and correlation coefficients

An attacker needs to know how expressive the PAP he found are, knowing the PAP intersections count and the weight. Additionally it is interesting for him, if instance tracking is meaningful at all. He can detect this using empirical data like the correlation between the PAP intersection and the precision to be expected and the ADS weight and the precision, respectively. The two graphs of the corresponding correlation coefficients are also plotted in the figure (see ρ(PAP intersections, precision) and ρ(weight', precision)[5]). It is obvious that there is a correlation between these properties:

- The higher the weight of an ADS the higher the precision of the retrieval. This validates the assumption that some ADS (and the generating instances) are more identifiable than others. ADS with low weight can be discarded from tracking.
- The higher the PAP intersection the lower the precision. This is a result of the greater anonymity set for such ADS/instances.

To visualise these results, two ADS are randomly selected, one with a high weight (156.01) and one with a low weight (87.58):

Table 4. ADS with weight 87.58 and retrieval precision 0.82

`<DATE> <TIME> <HOST>.dip.t-dialin.net <IP> Mozilla/4.0 (compatible; MSIE 5.5 Windows 98; Win 9x 4.90) HTTP/1.0 GET */*`

There is an obvious difference in the precision of retrieval of both ADS. This observation becomes clear by analysing the two ADS: Mozilla 4.0, here an alias for the Microsoft Internet Explorer, is a very popular Web browser used by most of the Web surfers. Also the presented name of the operating systems, Windows 98, is used by many (non-expert) users in general. Here each instance using this

[5] due to technical limitations ρ is indicated as ,,rho"

Table 5. ADS with weight 156.01 and retrieval precision 0.97

```
<DATE> <TIME> <HOST>.uni-hamburg.de <IP> Mozilla/4.76 [de]
(X11; U; Linux 2.2.10 i686) HTTP/1.0 GET image/gif, image/x-xbitmap,
image/jpeg, image/pjpeg, image/png iso-8859-1,*,utf-8 gzip de, ex-MX, es, en
```

configuration can be sure to appear with an often used configuration (resulting in a low retrieval precision) and to be hidden in a large anonymity set (leading to high number of intersections). Is has to be remarked further on, that this ADS does not give any additional information like language or media type preferences.

The other instance is using an operating system for professionals. This scores in a high term weight in the `User-Agent` field. While the media type field (`Accept`) is not very unusual, the `Accept-Language` header contain a seldom term like `es-mx` standing for ,,Spanish-Mexican". Such a configuration can be reidentified easily.

5 Countermeasures

It has been shown that instances are easier traceable if the variations in the ADS they generate are low (with respect to the selected HTTP fields and the value for Δsim). Considering the number of PAP intersections as a measure of anonymity of an instance, one can easily identify two countermeasures: Increasing the anonymity set and decreasing the relative similarity to the probe ADS. A solution suitable for each problem are anonymising proxies.

5.1 Proxies

A proxy is a service acting as an intermediary between a Web client and a Web server. Requests from the client are treated in the desired way and then forwarded to the server. Two kinds of proxies are applicable to reach or enlarge the anonymity of an instance and therefore to decrease the degree of assurance of found PAP for a probed ADS.

Anonymising Web Proxies Services for the anonymising of Web users are already existent on the Internet. The best known are *Anonymizer* [10] and *Rewebber* [11, 12]. Users can access Web pages directly via entering the intended URL in a Web form at each service or by configuring the service as their default proxy in the Web browser. The proxy acts as a middleman for the user. Both services are anonymising the HTTP header by exchanging the relevant HTTP header content with own text. This way an adversary logs a much higher number of ADS from such a service than a number of different ADS from the instances.

Today anonymising proxies are not used that much, therefore they can not be seen as a standard part of the Internet infrastructure.

5.2 Local Proxies

An approach to decrease the similarity is to vary header fields in each access to a Web page leading to very different sum of term weights in each access. This is only possible in a few header fields because many fields are determining the reaction of the server in an essential way (`Accept-Language` determines the language of a Web page sent back by the Web server, etc.). As an alternative, the existent header content can be extended by valid terms. In the case of `Accept-Language` (and most other HTTP header fields) this is possible, because header terms are considered with priority from the left to the right. Added extension can vary the header field (and change the similarity to the probe ADS) while the intended settings of a field are interpreted correctly.

The described functionality can be performed by a simple software running locally on a user's computer.

6 Conclusion

It has been shown that active attacks on a Web user's privacy are obvious and conspicious but also easy to defend by means of anonymising techniques.

The proposed passive attack uses access data that can easily be created. It shows a way and procedure how to measure a Web access by an instance, how to compare a probe ADS to other ADS, and how to track instances.
The quality of the results can be measured with the shown degree of assurance.

Furthermore, the attack can be applied in real-time using common computer hardware.

The attack shows the vulnerability of the privacy of Web users and motivates the use of anonymising techniques. Two simple countermeasures are proposed which are even applicable for the normal Web user.

References

1. Fielding, R., Gettys, J., et al, J.M.: Hypertext transfer protocol – http/1.1 (1999)
2. Gold, J., Ethier, D.: One perspective on collecting data on the web. CMC - Computer-Mediated Communication Magazine **9** (1997)
3. Shen, A.: Request for participation and comment from the electronic privacy information center (epic). (1999)
4. McGregor, G.: The ppp internet protocol control protocol (ipcp) (1992)
5. Salton, G.: Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley (1989)
6. Salton, G.: Automatic Information Organization and Retrieval. McGraw-Hill (1968)
7. Chaum, D.L.: Untraceable electronic mail, return addresses, and digital pseudonyms. In Rivest, R., ed.: Communication of the ACM. Volume 2. (1981) 84–88
8. Pfitzmann, A., Köhntopp, M.: Anonymity, unobservability, and pseudonymity - a proposal for terminology. International Workshop on Design Issues in Anonymity and Unobservability **LNCS 2009** (2000) 1–9

9. Berthold, O., Pfitzmann, A., Standtke, R.: The disadvantages of free mix routes and how to overcome them. International Workshop on Design Issues in Anonymity and Unobservability **LNCS 2009** (2000) 30–45
10. : Anonymizer. (http://www.anonymizer.com/)
11. : Rewebber. (http://www.rewebber.com/)
12. Demuth, T., Rieke, A.: On securing the anonymity of content providers in the world wide web. Proceedings of SPIE **3657** (1999) 494–502